

Introducción al Deep Learning

Ignacio Díaz Blanco

¿Qué es el aprendizaje profundo?

deep learning (DL)

Yosua Bengio: *El País*, 1 de febrero de 2018 (entrevista)

<https://goo.gl/vkndTo>

“Es un aprendizaje automático que permite a los ordenadores **aprender a partir de ejemplos** y construir sistemas inteligentes”



“... se distingue de otros aprendizajes en que los algoritmos usados se inspiran en las **redes de neuronas...**”

“Decimos que es **profundo** porque los cálculos se efectúan a través de diferentes **niveles de transformación** de datos, de distintas **capas, como nuestro cerebro**”



Colin Ware, 2008 “Visual thinking for design”

¿Qué es el aprendizaje profundo?

impacto futuro

«La inteligencia artificial tendrá el mismo impacto que la electricidad o el fuego»

- Sundar Pichai, CEO de Google, ha llevado a cabo esta afirmación en una entrevista que se emitirá próximamente en en la cadena estadounidense MSNBC



Sundar Pichai, CEO de Google - EFE



ABC TECNOLOGÍA
@abc_tecnologia

MADRID - 22/01/2018 12:11h - Actualizado: 23/01/2018 09:05h



abc.es

Sundar Pichai, CEO de Google, enero de 2018

<https://goo.gl/orH71V>

Artículo en Forbes

“10 Ways Machine Learning Is Revolutionizing Manufacturing”

JUN 26, 2016 @ 08:17 PM

<https://goo.gl/BzYWZa>


Algunas ideas

- Incrementar capacidad de producción 20%
- Reducir consumo de material 4%
- Aportar datos más relevantes → ajustar acciones (financiación, operaciones, etc.)
- Mejoras en mantenimiento: preventivo, predictivo
- Monitorización de la condición → mejora en la eficiencia de los procesos
- Control de calidad: factores que afectan calidad (productos, empresa: KPI's)

¿Dónde está el Deep Learning?

En el bolsillo...

chips dedicados en dispositivos móviles



en.wikipedia.org

WIKIPEDIA The Free Encyclopedia

Apple A17

Article Talk Read Edit View history Tools

From Wikipedia, the free encyclopedia

The **Apple A17 Pro** is a 64-bit ARM-based system on a chip (SoC) designed by **Apple Inc.** and manufactured by **TSMC**.^[4] It is used in the **iPhone 15 Pro** and **iPhone 15 Pro Max** models only.^{[3][5]}

Design

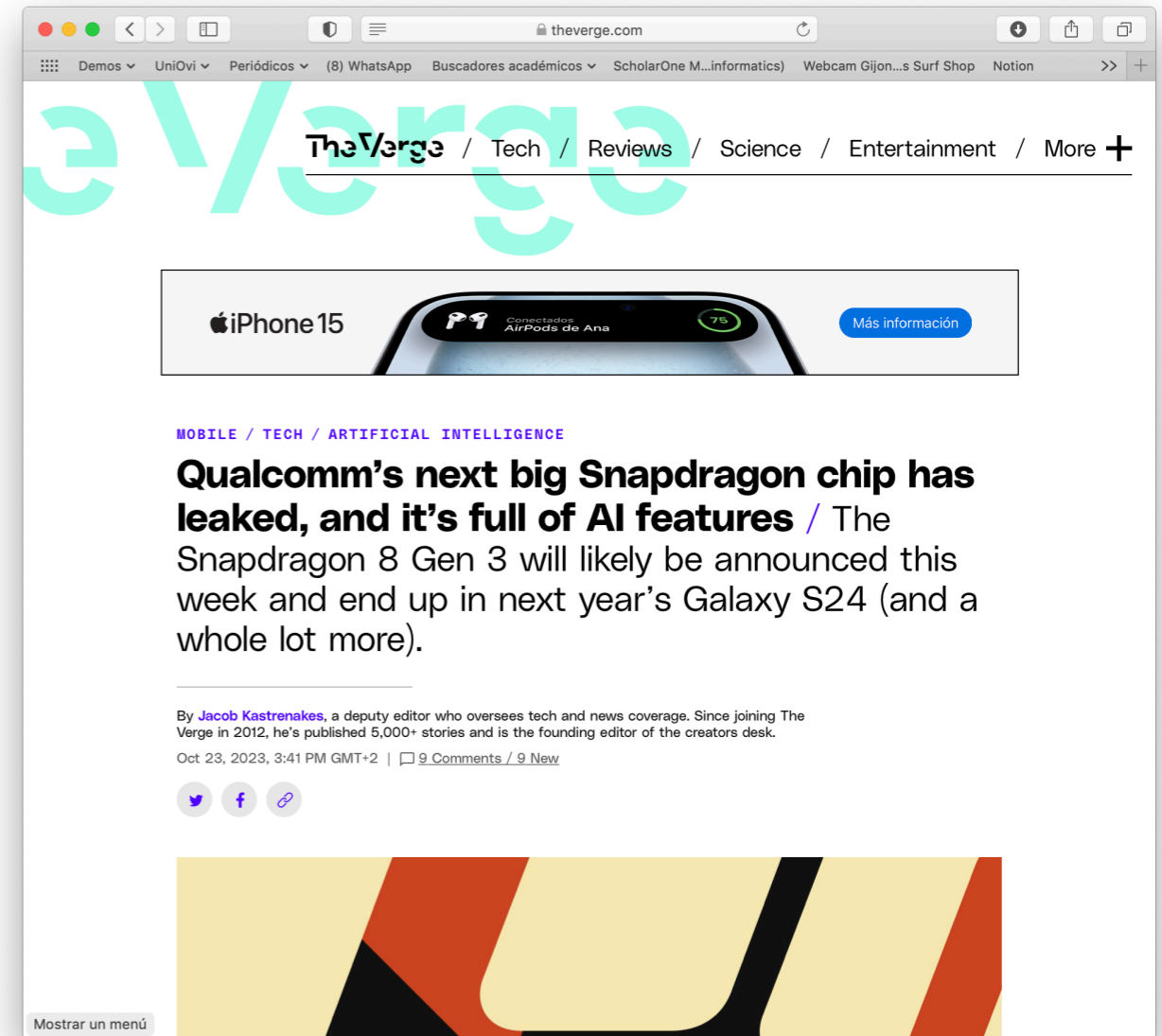
The Apple A17 Pro features an Apple-designed 64-bit six-core **CPU** with two high-performance cores running at 3.78 GHz, and four energy-efficient cores running at 2.11 GHz.^[2] Apple claims the new high-performance cores are 10% faster due to its improved branch prediction, and wider decode & execution engines, and the new energy-efficient cores are faster and 3x more efficient than the competition.^[5] The amount of RAM has increased from 6GB to 8GB.^[6]

The A17 Pro integrates a new Apple-designed six-core **GPU**, which Apple claims is 20% faster and their biggest redesign in the history of Apple GPUs, with added hardware accelerated **ray tracing** and mesh shading support. The 16-core Neural Engine is now capable of 35 trillion operations per second. The A17 Pro also added support for **AV1** decoding and **USB 3.2 Gen 2** (est. up to 10Gbit/s / 1.25GBps).^[7] The A17 Pro contains 19 billion **transistors**, a 19% increase from the **A16**'s **transistor count** of 16 billion, and is fabricated by **TSMC** on their **3 nm N3B** process.^[5]

Products that include the Apple A17 Pro

- iPhone 15 Pro & 15 Pro Max

Apple A17 Pro	
General information	
Launched	September 12, 2023
Designed by	Apple Inc.
Common manufacturer(s)	TSMC ^[1]
Product code	APL1V02
Max. CPU clock rate	to 3.78 GHz ^[2]
Cache	
L2 cache	16 MB (performance cores) 4 MB (efficiency cores)
L4 cache	24 MB (system level cache)
Architecture and classification	
Application	Mobile
Technology node	3 nm (N3B)
Instruction set	AArch64
Physical specifications	
Transistors	19 billion
Cores	6 (4 efficiency, 2 performance) ^[3]
Memory (RAM)	8GB
GPU(s)	Apple-designed 6-core GPU ^[3]
History	
Predecessor(s)	Apple A16



theverge.com

The Verge / Tech / Reviews / Science / Entertainment / More +

iPhone 15

Qualcomm's next big Snapdragon chip has leaked, and it's full of AI features / The Snapdragon 8 Gen 3 will likely be announced this week and end up in next year's Galaxy S24 (and a whole lot more).

MOBILE / TECH / ARTIFICIAL INTELLIGENCE

By **Jacob Kastrenakes**, a deputy editor who oversees tech and news coverage. Since joining The Verge in 2012, he's published 5,000+ stories and is the founding editor of the creators desk.

Oct 23, 2023, 3:41 PM GMT+2 | 9 Comments / 9 New

En cualquier dispositivo...

PC's, tablets,... todo lo que se conecte a internet

Reconocimiento de voz

Siri, Alexa

Recomendadores

Netflix, Amazon, Google, X (Twitter)

Etiquetado de imágenes

Google, Facebook

Detección de contenido inapropiado en vídeos de YouTube

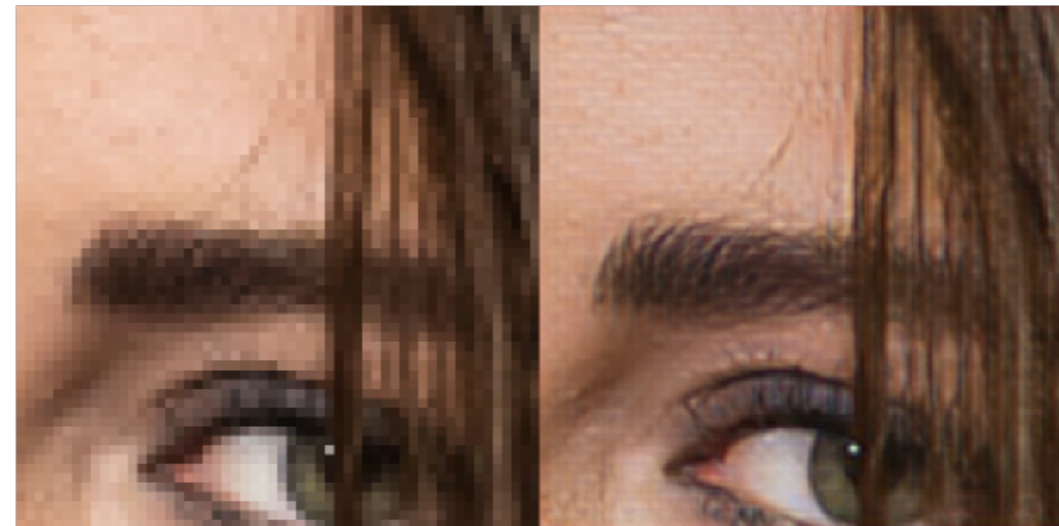
- subidas de vídeos: **400 h/min**
- Sistema IA:
identificó **98%** de **150.000** archivos retirados
- Fuerza de trabajo:
180.000 personas trabajando **40h/semana**
- Sin embargo, YouTube anunciaba a finales de 2017 que también **reforzaría su equipo humano**



fuente: abc tecnología 06/02/2018, <https://goo.gl/bdxfd1>

Superresolución de imágenes

generative adversarial networks, GAN



<https://github.com/alexjc/neural-enhance>

¿El truco? la red “alucina”, se inventa los detalles que faltan
basándose en miles de ejemplos aprendidos

Cámaras Ricoh PENTAX

Modo "AUTO"

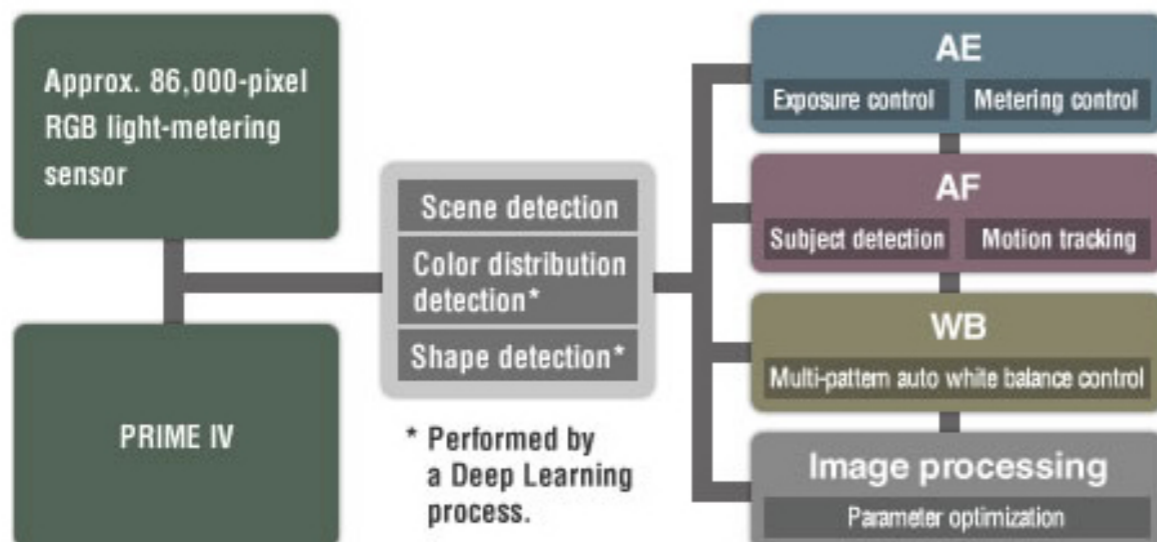


basándose en datos de sensor RGB
(~ 86000 píxeles)

análisis de escenas

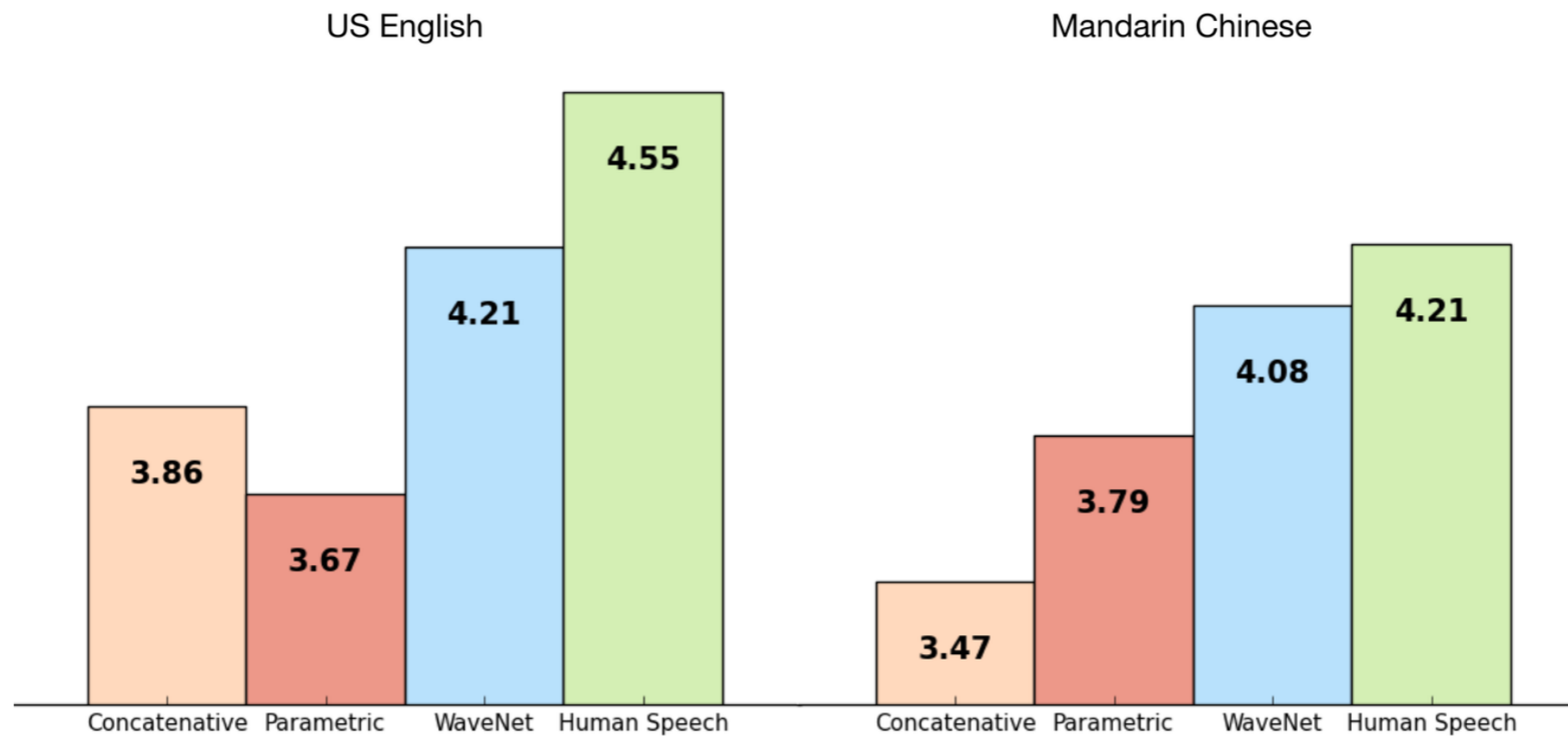
optimiza parámetros de exposición

fuente: página web de Ricoh, <https://goo.gl/BSJAaf>



Máquinas que hablan

Síntesis de voz: DeepMind/WaveNet



fuelle: <https://deepmind.com/blog/wavenet-generative-model-raw-audio/>

En la página web, se puede comparar el resultado de WaveNet con otros métodos

$$\mathcal{L}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{\text{contenido}}(\vec{p}, \vec{x}) + \beta \vec{L}_{\text{estilo}}(\vec{a}, \vec{x})$$



A Neural Algorithm of Artistic Style

Leon A. Gatys,^{1,2,3*} Alexander S. Ecker,^{1,2,4,5} Matthias Bethge^{1,2,4}

¹Werner Reichardt Centre for Integrative Neuroscience
and Institute of Theoretical Physics, University of Tübingen, Germany

²Bernstein Center for Computational Neuroscience, Tübingen, Germany

³Graduate School for Neural Information Processing, Tübingen, Germany

⁴Max Planck Institute for Biological Cybernetics, Tübingen, Germany

⁵Department of Neuroscience, Baylor College of Medicine, Houston, TX, USA

*To whom correspondence should be addressed; E-mail: leon.gatys@bethgelab.org

In fine art, especially painting, humans have mastered the skill to create unique visual experiences through composing a complex interplay between the content and style of an image. Thus far the algorithmic basis of this process is unknown and there exists no artificial system with similar capabilities. However, in other key areas of visual perception such as object and face recognition near-human performance was recently demonstrated by a class of biologically inspired vision models called Deep Neural Networks.^{1,2} Here we introduce an artificial system based on a Deep Neural Network that creates artistic images of high perceptual quality. The system uses neural representations to separate and recombine content and style of arbitrary images, providing a neural algorithm for the creation of artistic images. Moreover, in light of the striking similarities between performance-optimised artificial neural networks and biological vision,³⁻⁷ our work offers a path forward to an algorithmic understanding of how humans create and perceive artistic imagery.

1

arXiv:1508.06576v2 [cs.CV] 2 Sep 2015

fuente: <https://github.com/jcjohnson/neural-style>

fuente: <https://arxiv.org/abs/1508.06576>

Conducción autónoma (reinforcement learning)



fuelle YouTube: <https://youtu.be/DeE8tntTSPi>

Proyecto DeepDrive en Berkeley

<https://deepdrive.berkeley.edu/>



Google search results for "inteligencia artificial hacienda".

Diario de Noticias de Álava
<https://www.noticiasdealava.eus> › alava › 2023/11/22
La Hacienda de Álava recurrirá a la inteligencia artificial ...
 hace 2 días — La **Hacienda** de Álava recurrirá a herramientas vinculadas con la **inteligencia artificial** para luchar contra el fraude fiscal, como es el caso de un sistema ...

Libre Mercado
<https://www.libremercado.com> › Libre Mercado
De la inspección física a los algoritmos
 31 oct 2023 — **Hacienda** considera que ha tenido una "experiencia satisfactoria en la utilización de la **inteligencia artificial** como forma de proporcionar información ...

Antifraucv
<https://www.antifraucv.es> › inteligencia-artificial-y-luc...
Inteligencia Artificial y lucha contra la corrupción
 hace 3 días — La **inteligencia artificial** puede ayudar a combatir la corrupción al detectar irregularidades y conflictos de interés mediante el análisis masivo de datos. Para ...

FiscalBlog
<https://fiscalblog.es> › Derechos fundamentales
¿Puede la Agencia Tributaria utilizar la Inteligencia Artificial?
 29 oct 2023 — 117 RGAT, permitan al Ministro de **Hacienda** establecer con carácter general una obligación allí donde el art. 96.2 LGT establece un derecho.» Este ...

(fecha consulta 24/11/2023)

Google search results for "inteligencia artificial justicia juzgado".

UNESCO
<https://www.unesco.org> › rule-law › mooc-judges
La IA y el Estado de derecho: Fortalecimiento ...
 8 nov 2023 — Los sistemas judiciales a nivel mundial utilizan la **Inteligencia artificial (IA)** para analizar grandes cantidades de datos jurídicos con el objetivo de ayudar a ...

Europa Press
<https://www.europapress.es> › madrid › noticia-juzgado...
Los juzgados madrileños dispondrán a final de año de un ...
 25 oct 2023 — Los **juzgados** de la Comunidad de Madrid dispondrán "a finales de años" de un buscador basado en **Inteligencia Artificial (IA)** para poder consultar toda la ...

El Derecho
<https://elderecho.com> › justicia-pone-en-marcha-un-pr...
Justicia pone en marcha un programa de hyper ...
 10 nov 2023 — Además, a través del uso de la RPA -automatización robótica de procesos-, y la combinación con técnicas de **inteligencia artificial ... juzgado** para lograr su ...

Europa Press
<https://www.europapress.es> › economia › noticia-minis...
El Ministerio de Justicia y NTT Data colaboran para ...
 10 nov 2023 — ... **inteligencia artificial (IA)** para mejorar la eficiencia de los órganos ... **juzgado** para lograr su adaptación digital y ser parte fundamental del proyecto ...

Derecho práctico
<https://www.derechopractico.es> › el-ministerio-de-justi...
El Ministerio de Justicia pone en marcha un programa de ...
 10 nov 2023 — A través de la combinación de diferentes tecnologías basadas en robotización e **inteligencia artificial**, este programa fomentará la eficiencia de los órganos ...

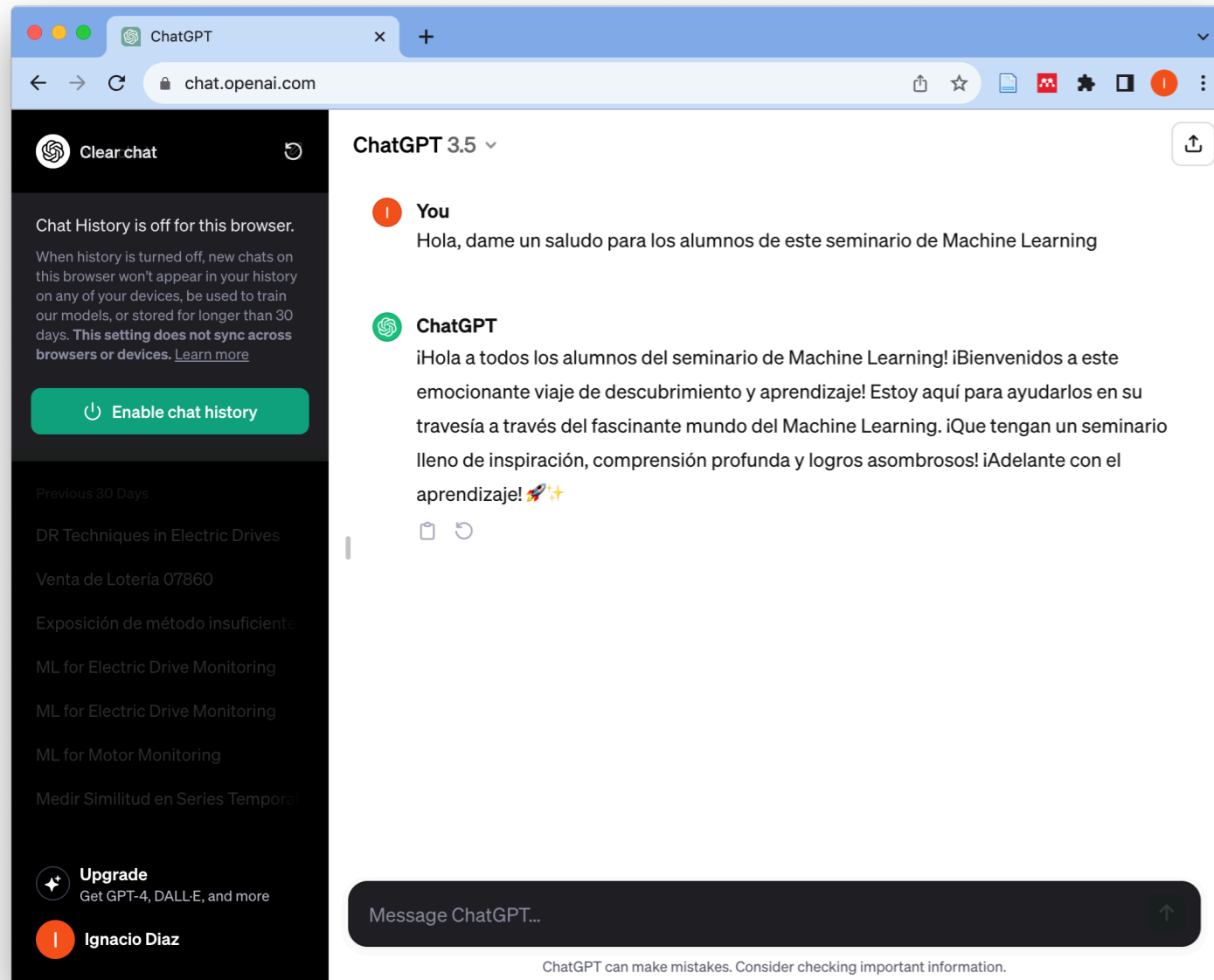
Ámbito Jurídico
<https://www.ambitojuridico.com> › columnista-impresso
El Ministerio de Justicia pone en marcha un programa de ...
 10 nov 2023 — A través de la combinación de diferentes tecnologías basadas en robotización e **inteligencia artificial**, este programa fomentará la eficiencia de los órganos ...

(fecha consulta 24/11/2023)

Modelos de Language

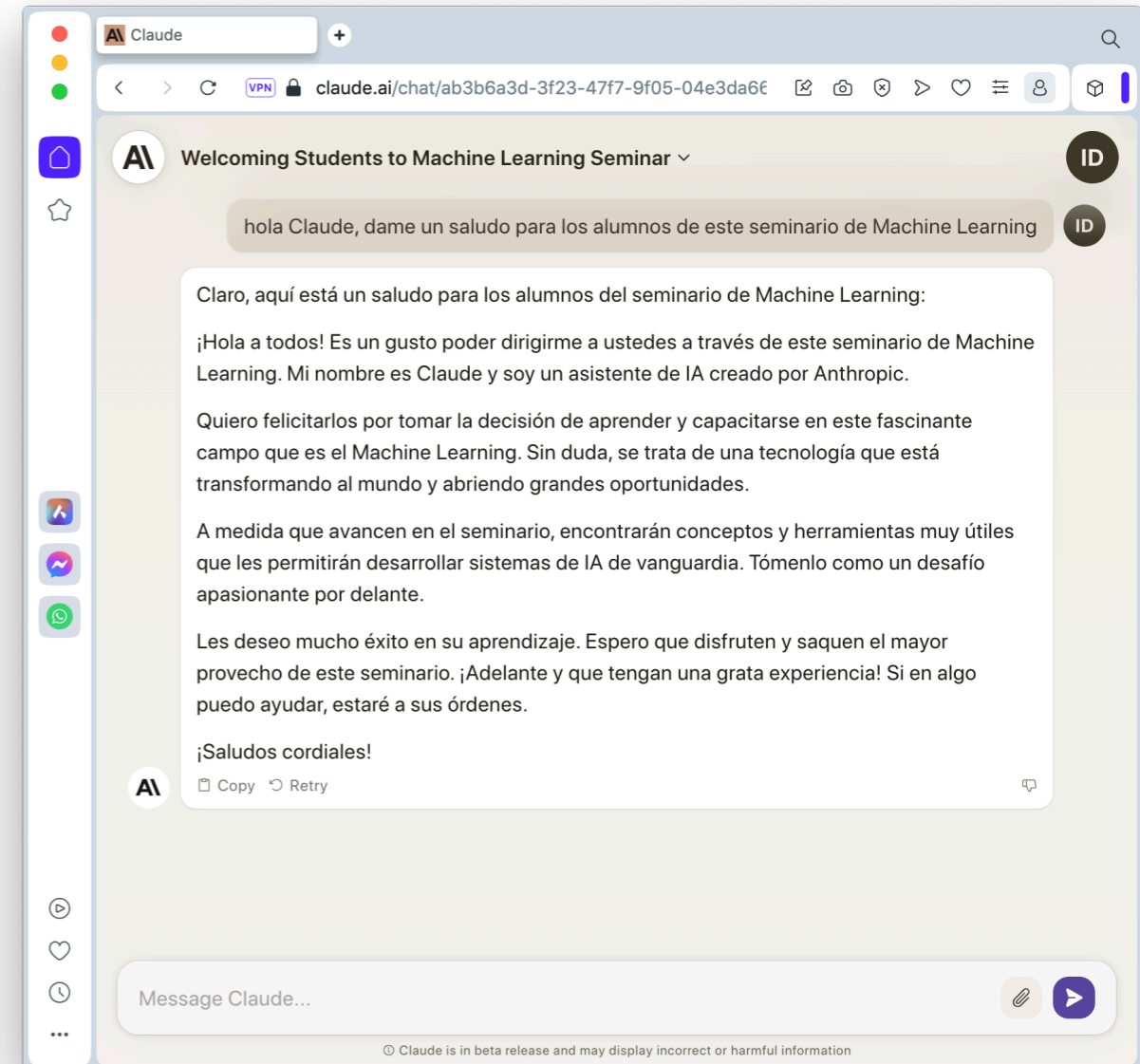
chatGPT, claude.ai, perplexity.ai, ...

ChatGPT



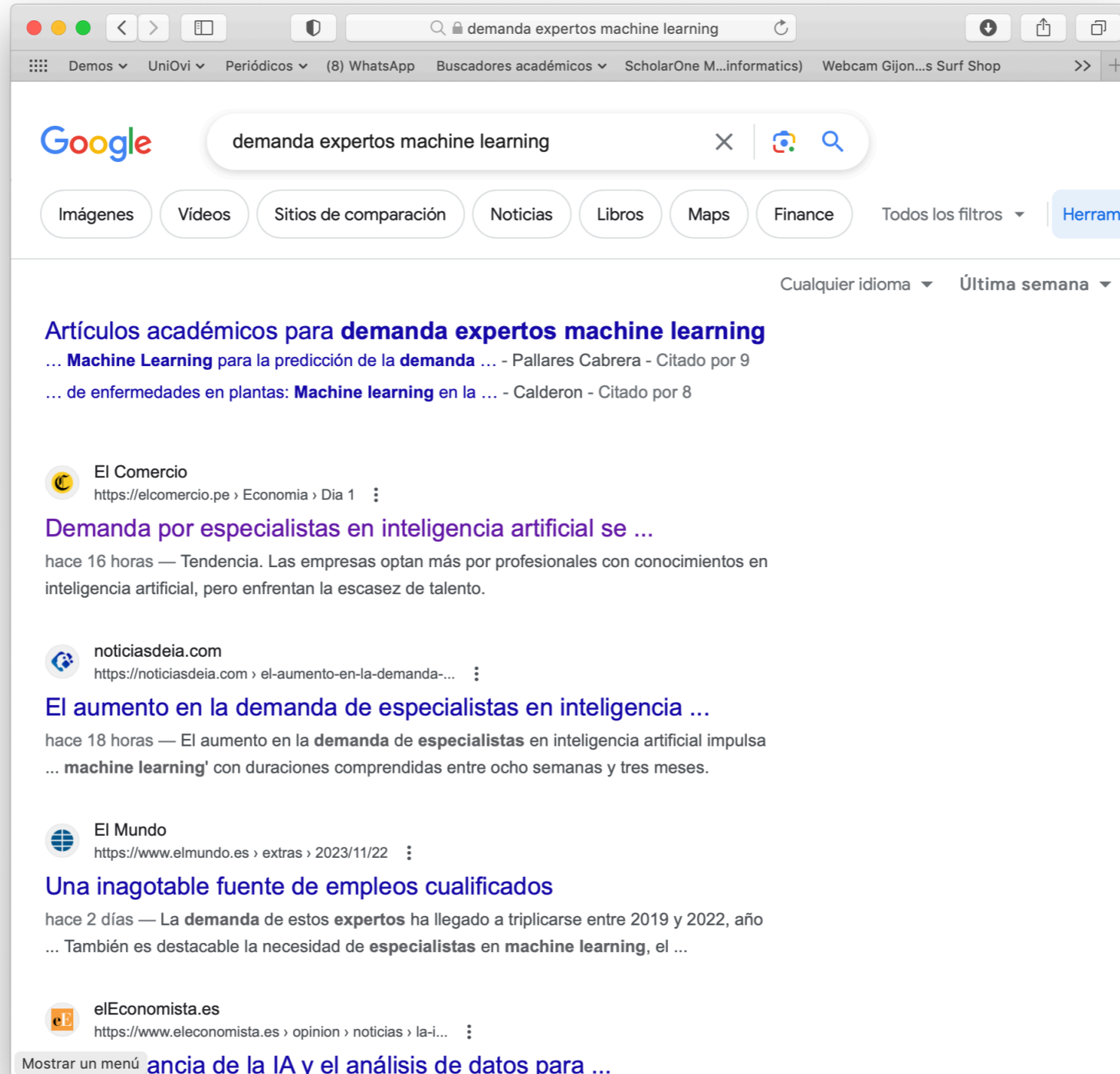
The screenshot shows the ChatGPT web interface in a browser. The URL is chat.openai.com. The chat is titled 'ChatGPT 3.5'. The user's message is: 'Hola, dame un saludo para los alumnos de este seminario de Machine Learning'. The AI's response is: '¡Hola a todos los alumnos del seminario de Machine Learning! ¡Bienvenidos a este emocionante viaje de descubrimiento y aprendizaje! Estoy aquí para ayudarlos en su travesía a través del fascinante mundo del Machine Learning. ¡Que tengan un seminario lleno de inspiración, comprensión profunda y logros asombrosos! ¡Adelante con el aprendizaje! 🚀✨'. A sidebar on the left shows chat history with a note that it is off for this browser. At the bottom, there is a text input field with 'Message ChatGPT...' and a disclaimer: 'ChatGPT can make mistakes. Consider checking important information.'

Claude



The screenshot shows the Claude web interface in a browser. The URL is claude.ai/chat/ab3b6a3d-3f23-47f7-9f05-04e3da6€. The chat is titled 'Welcoming Students to Machine Learning Seminar'. The user's message is: 'hola Claude, dame un saludo para los alumnos de este seminario de Machine Learning'. The AI's response is: 'Claro, aquí está un saludo para los alumnos del seminario de Machine Learning: ¡Hola a todos! Es un gusto poder dirigirme a ustedes a través de este seminario de Machine Learning. Mi nombre es Claude y soy un asistente de IA creado por Anthropic. Quiero felicitarlos por tomar la decisión de aprender y capacitarse en este fascinante campo que es el Machine Learning. Sin duda, se trata de una tecnología que está transformando al mundo y abriendo grandes oportunidades. A medida que avancen en el seminario, encontrarán conceptos y herramientas muy útiles que les permitirán desarrollar sistemas de IA de vanguardia. Tómenlo como un desafío apasionante por delante. Les deseo mucho éxito en su aprendizaje. Espero que disfruten y saquen el mayor provecho de este seminario. ¡Adelante y que tengan una grata experiencia! Si en algo puedo ayudar, estaré a sus órdenes. ¡Saludos cordiales!'. At the bottom, there is a text input field with 'Message Claude...' and a disclaimer: 'Claude is in beta release and may display incorrect or harmful information.'

Trabajo en deep learning expertos en DL muy demandados ...



demanda expertos machine learning

Google

demanda expertos machine learning

Imágenes Vídeos Sitios de comparación Noticias Libros Maps Finance Todos los filtros Herramientas

Cualquier idioma Última semana

Artículos académicos para demanda expertos machine learning

... **Machine Learning** para la predicción de la **demanda** ... - Pallares Cabrera - Citado por 9

... de enfermedades en plantas: **Machine learning** en la ... - Calderon - Citado por 8

El Comercio
https://elcomercio.pe › Economía › Día 1

Demanda por especialistas en inteligencia artificial se ...
hace 16 horas — Tendencia. Las empresas optan más por profesionales con conocimientos en inteligencia artificial, pero enfrentan la escasez de talento.

noticiasdeia.com
https://noticiasdeia.com › el-aumento-en-la-demanda-...

El aumento en la demanda de especialistas en inteligencia ...
hace 18 horas — El aumento en la **demanda** de **especialistas** en inteligencia artificial impulsa ... **machine learning** con duraciones comprendidas entre ocho semanas y tres meses.

El Mundo
https://www.elmundo.es › extras › 2023/11/22

Una inagotable fuente de empleos cualificados
hace 2 días — La **demanda** de estos **expertos** ha llegado a triplicarse entre 2019 y 2022, año ... También es destacable la necesidad de **especialistas** en **machine learning**, el ...

elEconomista.es
https://www.eleconomista.es › opinion › noticias › la-i-...

Mostrar un menú **ancia de la IA y el análisis de datos para ...**

(fecha consulta 24/11/2023)

¿Qué son las redes neuronales?

Breve historia de las RN

línea de tiempo

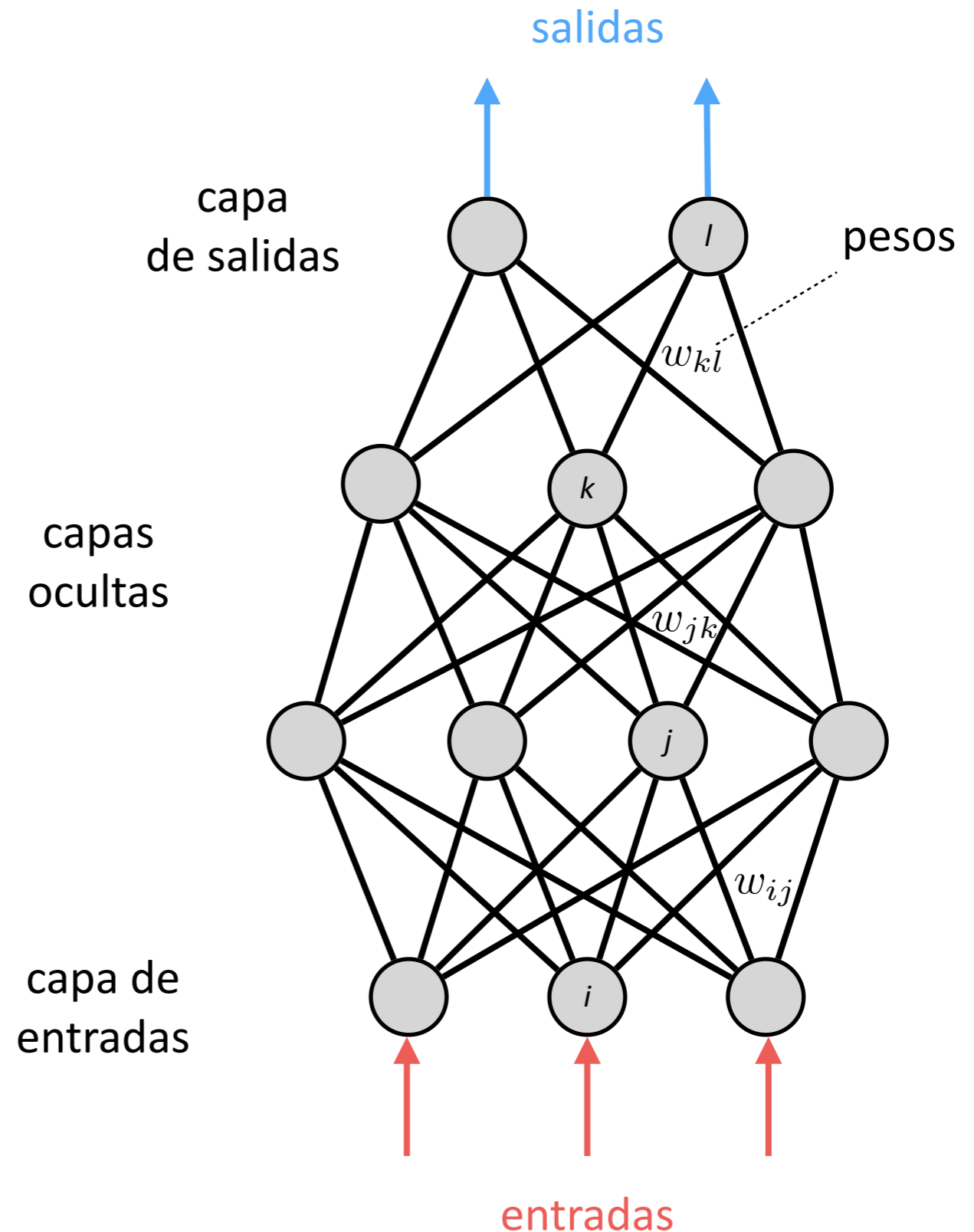
- Antecedentes**
- 16xx 1676 regla de la cadena (Leibniz)
1696 regla de la cadena (L'Hopital)
- 18xx 1847 optimización descenso gradiente (Cauchy)
- 1940 **Primeros modelos (lineales)**
1943 Mc Culloch & Pitts
1949 Regla aprendizaje Hebb
- 1950 **"Golden Years" (1956-1974)**
1956 conferencia Dartmouth: nace la IA
- 1960 ★ 1958 Perceptrón lineal (Rosenblatt)
1960 Adaline (Widrow y Hoff)
1969 Problema XOR en modelos lineales (Minsky Papert)
- 1970 **"first AI winter" (1974-1980)**
- 1980 **"Boom" (1980-1987)**
★ 1980 neocognitrón, primera CNN (Fukushima, 1980)
1981 aplicación regla cadena a NN (Werbos, 1981)
1985 backpropagation (LeCun, 1985)
★ 1985 backpropagation (Parker, 1985)
1986 backpropagation (Rummelhart et al, 1986)
1986 representación distribuida (Hinton, 1986)
1989 Entrenan redes profundas, pero en tiempos inviables
- 1990 **"second AI winter" (1987-1993)**
1995 SVM (Vapnik, 1995)
★ 1997 LSTM (Hochreiter, Schmidhuber, 1997)
★ 1998 LeNet, convolucional de 7 capas (LeCun, etc.)
- 2000 2000 Explosión de trabajos en SVM, SVR
- Deep learning**
- ★ 2006 Primera red profunda (Hinton, 2006)
2011 Rectified Linear Units, ReLu (Jarrett et al. 2009)
2012 AI consigue niveles humanos en reconocimiento
2013 Google y Facebook entran en DL
★ 2015 Google libera TensorFlow
2016 AlphaGo (Deepmind) vence a Lee Sedol
2017 AlphaGo Zero aprende por refuerzo

★ → hitos relacionados con deep learning

Las RN son algoritmos formados por unidades básicas denominadas **neuronas**

Organizadas en **capas** de varias neuronas

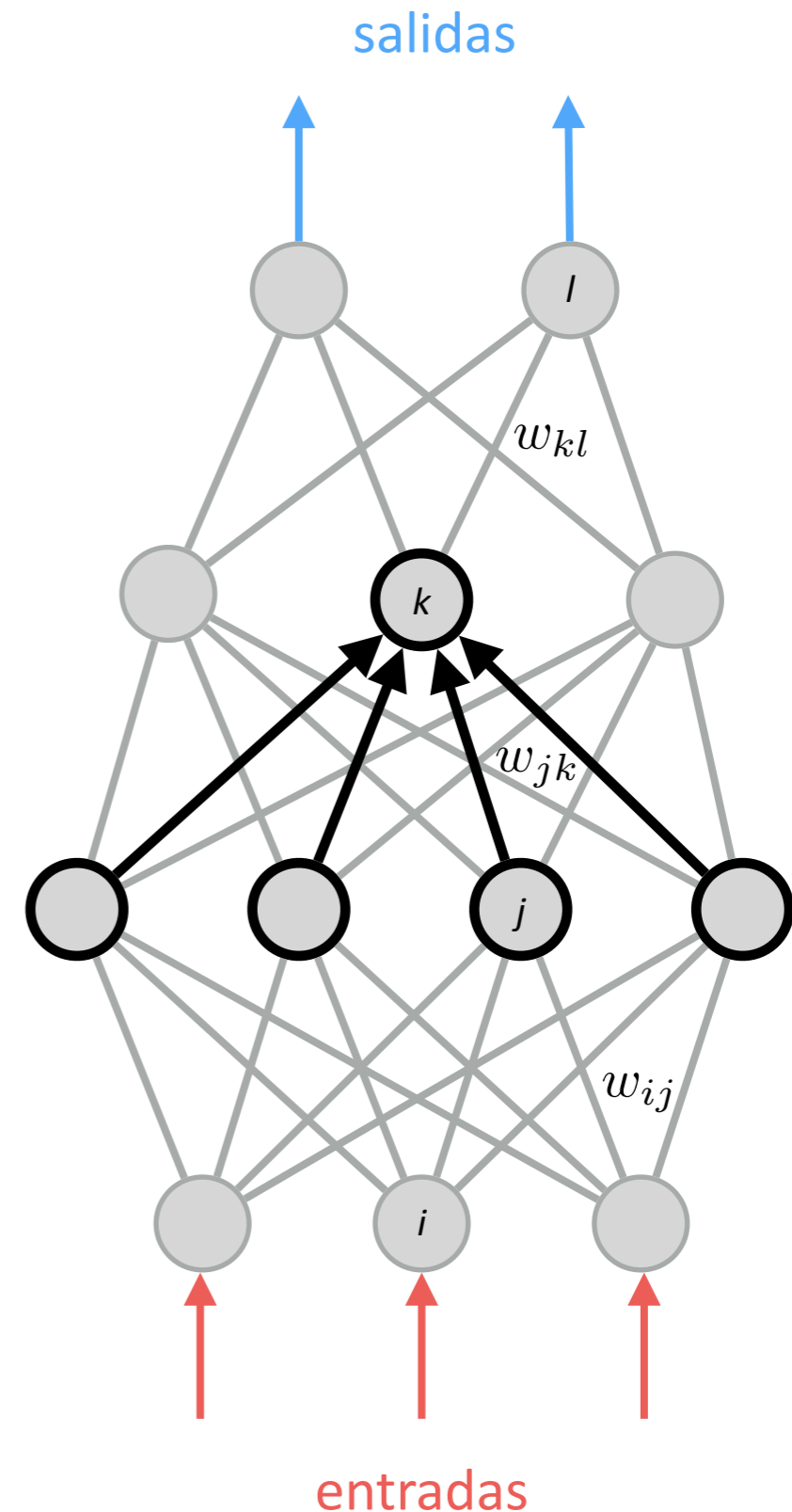
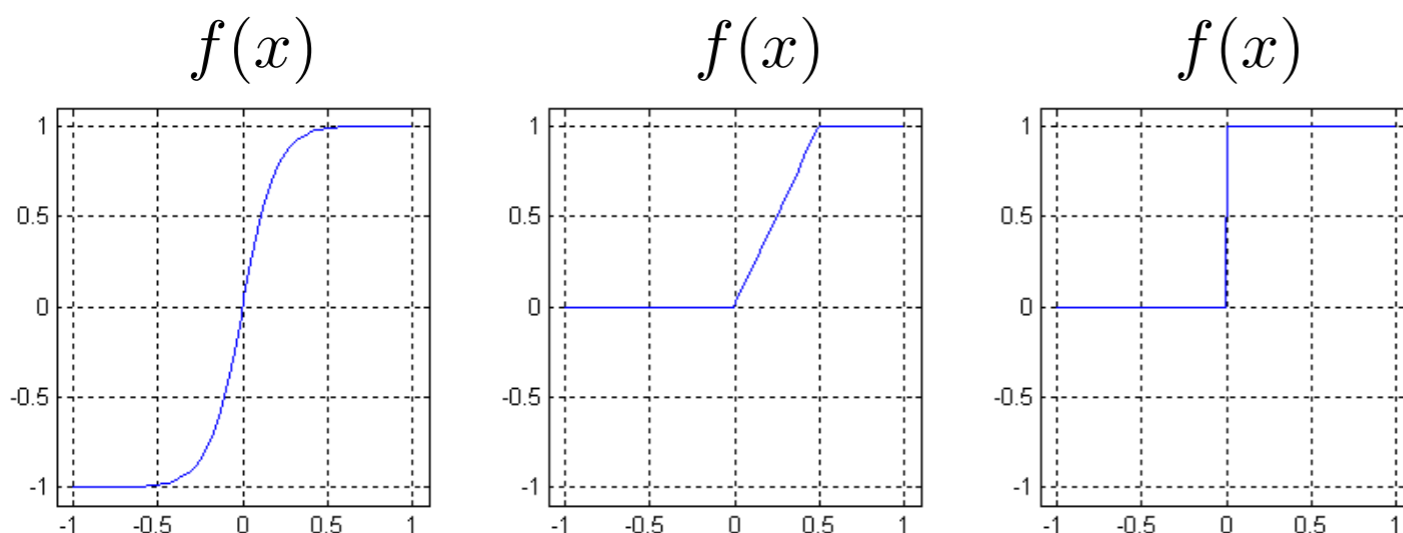
Las salidas de una capa son entradas a la capa siguiente



Cada neurona realiza
un procesamiento simple no lineal
(puede ser lineal, como caso particular)

$$a_k = f\left(\sum_j w_{jk} a_j\right)$$

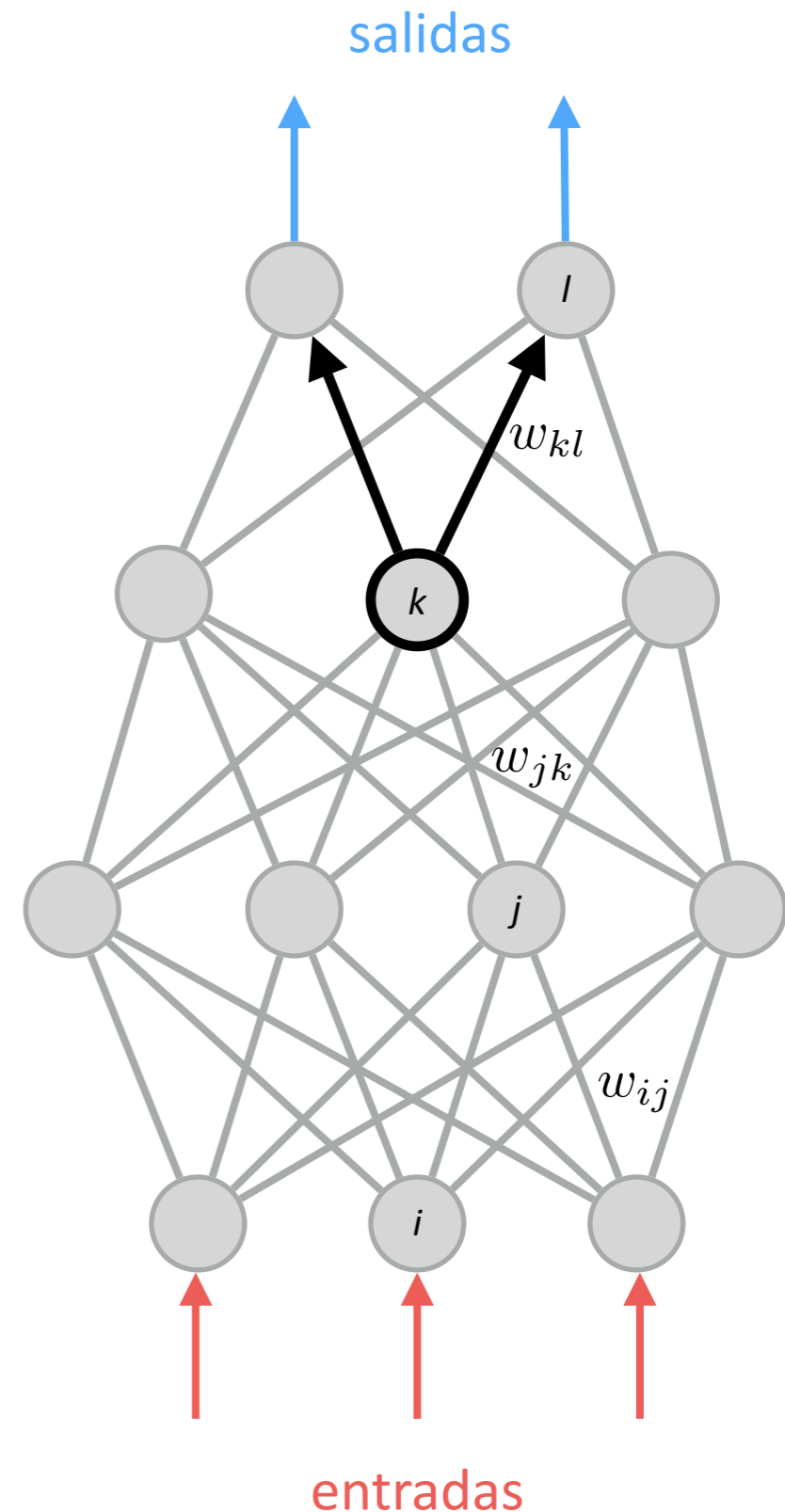
la **función de activación** f aporta la no linealidad...



El resultado de cada neurona se transfiere a otras neuronas

... esas neuronas vuelven a realizar el procesamiento descrito

... y así sucesivamente hasta producir una o varias salidas



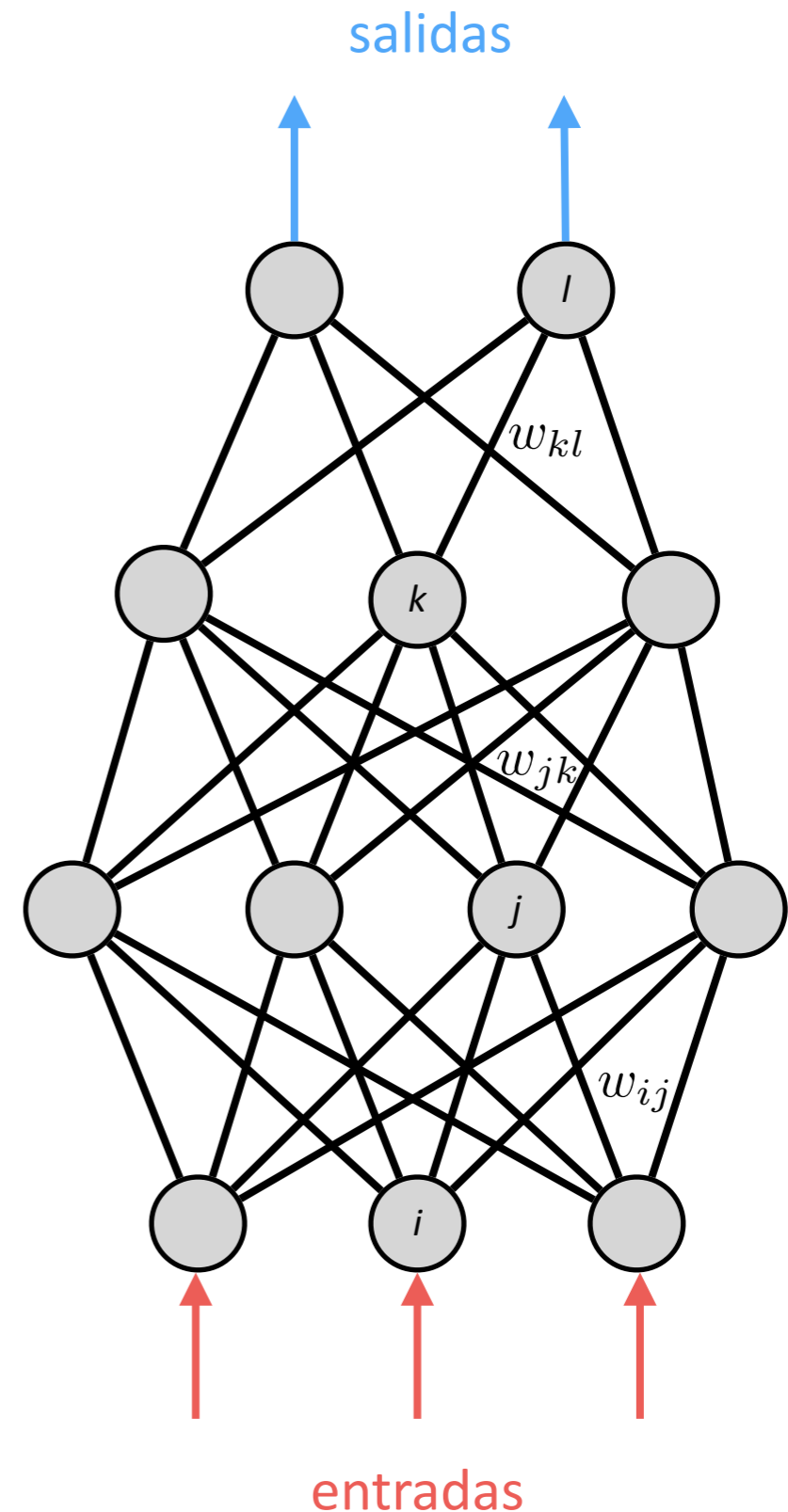
La red neuronal resulta ser una **composición de funciones**

$$a_l = f\left(\sum_k w_{kl} f\left(\sum_j w_{jk} a_j\right)\right)$$

$$a_l = f\left(\sum_k w_{kl} f\left(\sum_j w_{jk} f\left(\sum_j w_{rj} \dots\right)\right)\right)$$

$$\text{salidas} = f(g(h(i(\dots r(\text{entradas}) \dots))))$$

$$\text{salidas} = (f \circ g \circ h \circ i \dots \circ r)(\text{entradas})$$



La red neuronal resulta ser una **composición de funciones**

$$\text{salidas} = f(g(h(i(\dots r(\text{entradas}) \dots))))$$



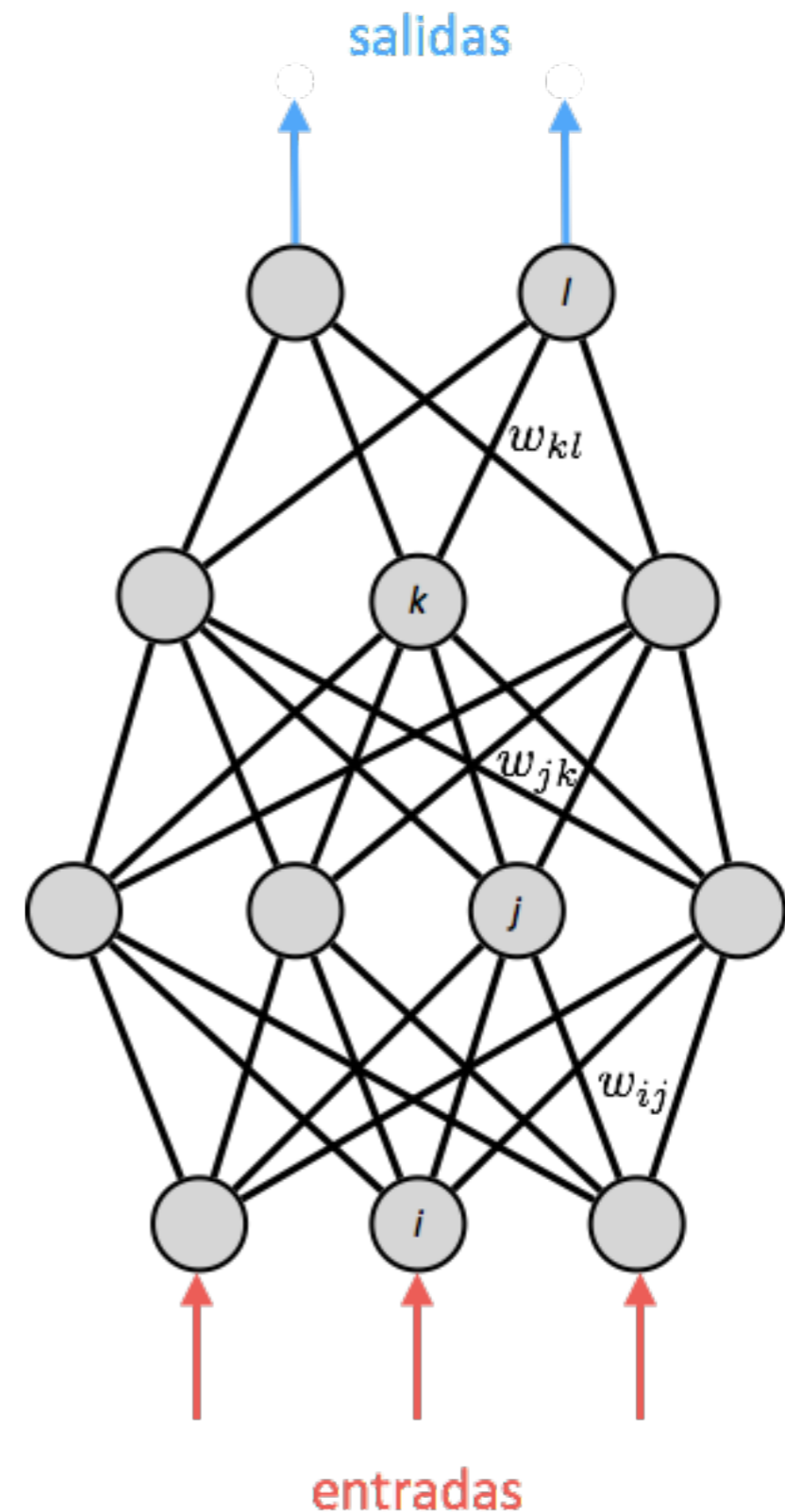
$$\text{salidas} = (f \circ g \circ h \circ i \dots \circ r)(\text{entradas})$$



$$\text{salidas} = F(\text{entradas}, \text{pesos})$$

matemáticamente:

$$\mathbf{y} = F(\mathbf{x}, \mathbf{W})$$



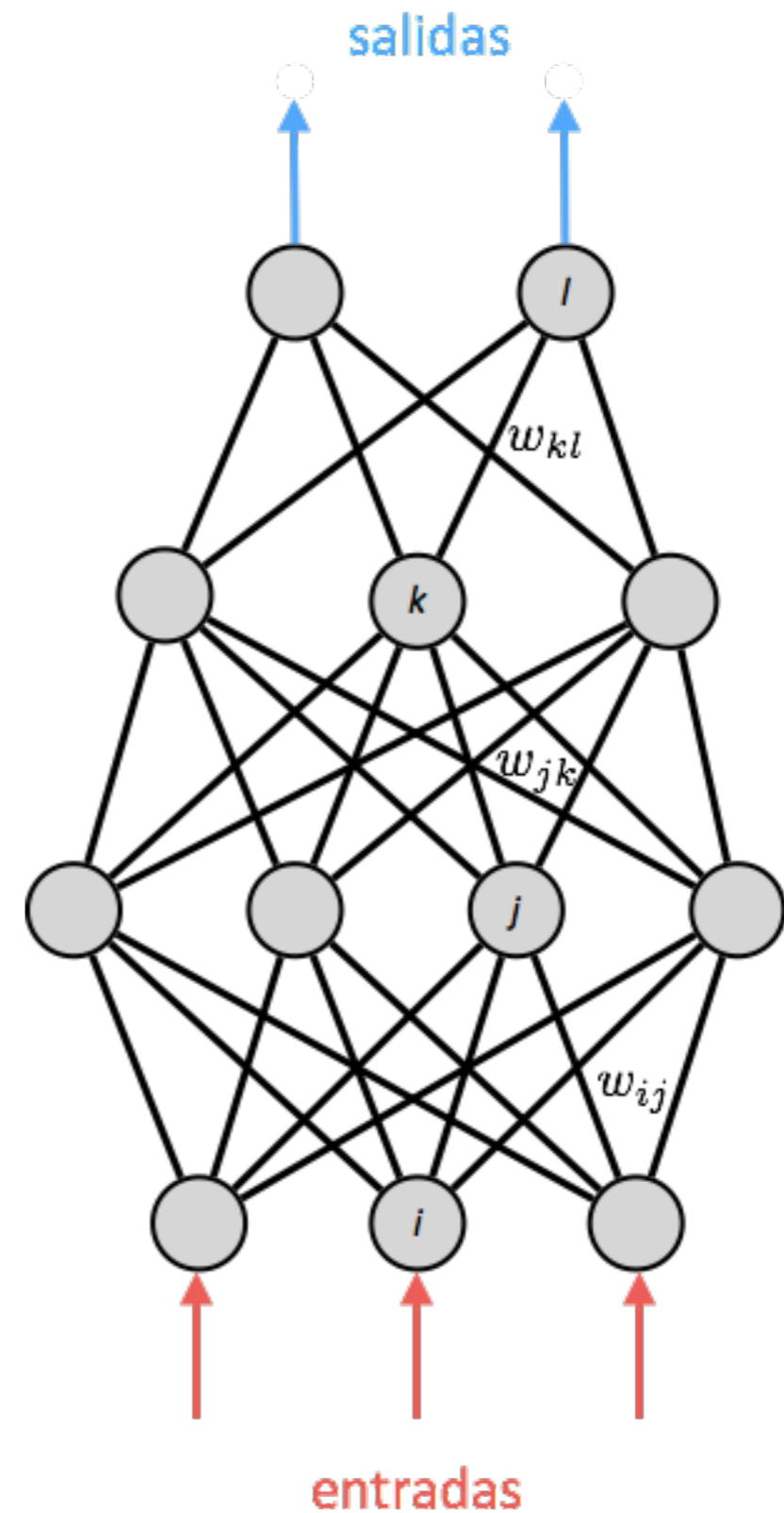
función compleja
resultante de composición
de funciones simples

$$\mathbf{y} = F(\mathbf{x}, \mathbf{W})$$

parámetros:
los pesos de todas
las neuronas

Los pesos W
permiten moldear/ajustar F
a casi cualquier función

$$\mathbf{y} = f(\mathbf{x})$$

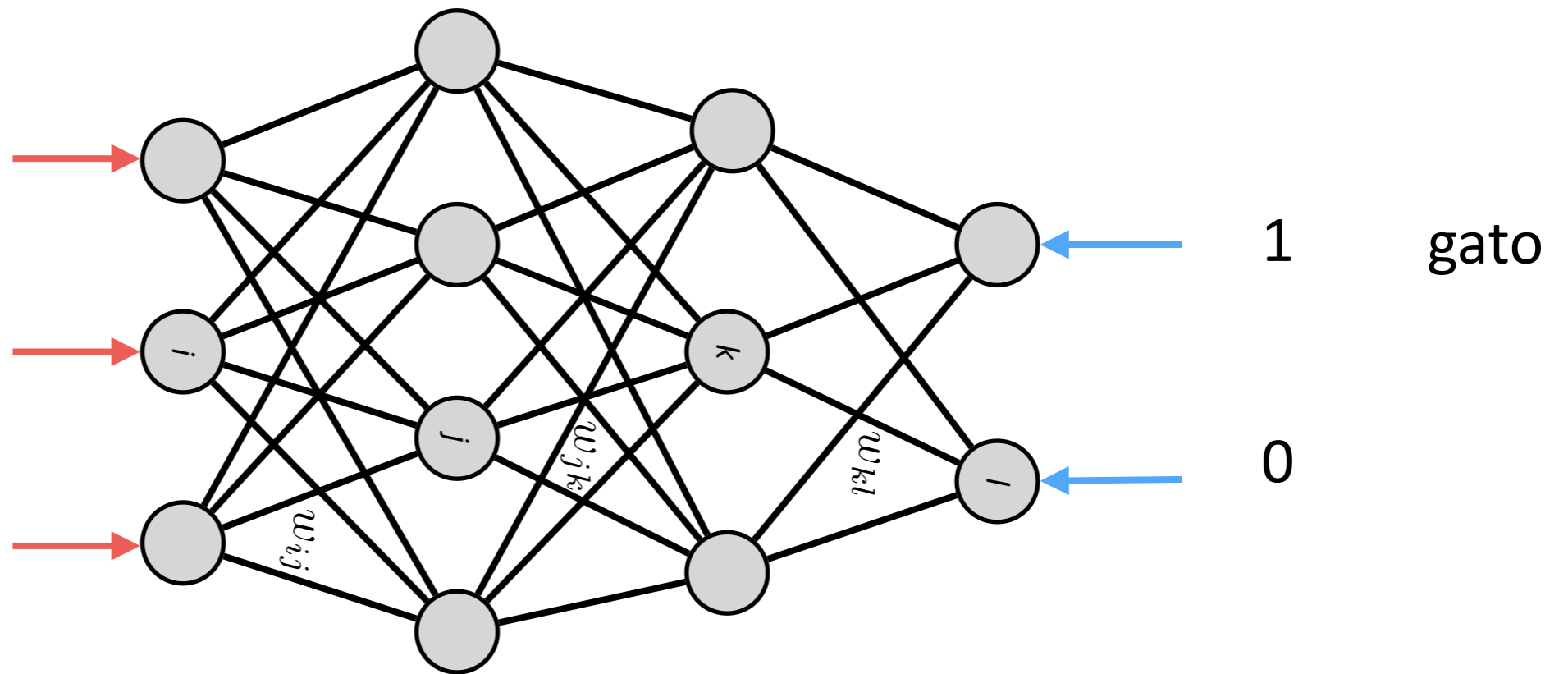


Aprendizaje a partir de ejemplos

Redes neuronales

Aprendizaje a partir de ejemplos

Una RN es capaz de **aprender a partir de ejemplos** ajustando los pesos W para minimizar un error
 → problema de optimización



datos $\{\mathbf{x}_i, \mathbf{y}_i\}$

clasificación: $\mathbf{y}_i = \{(0, 1), (1, 0), \dots\}$

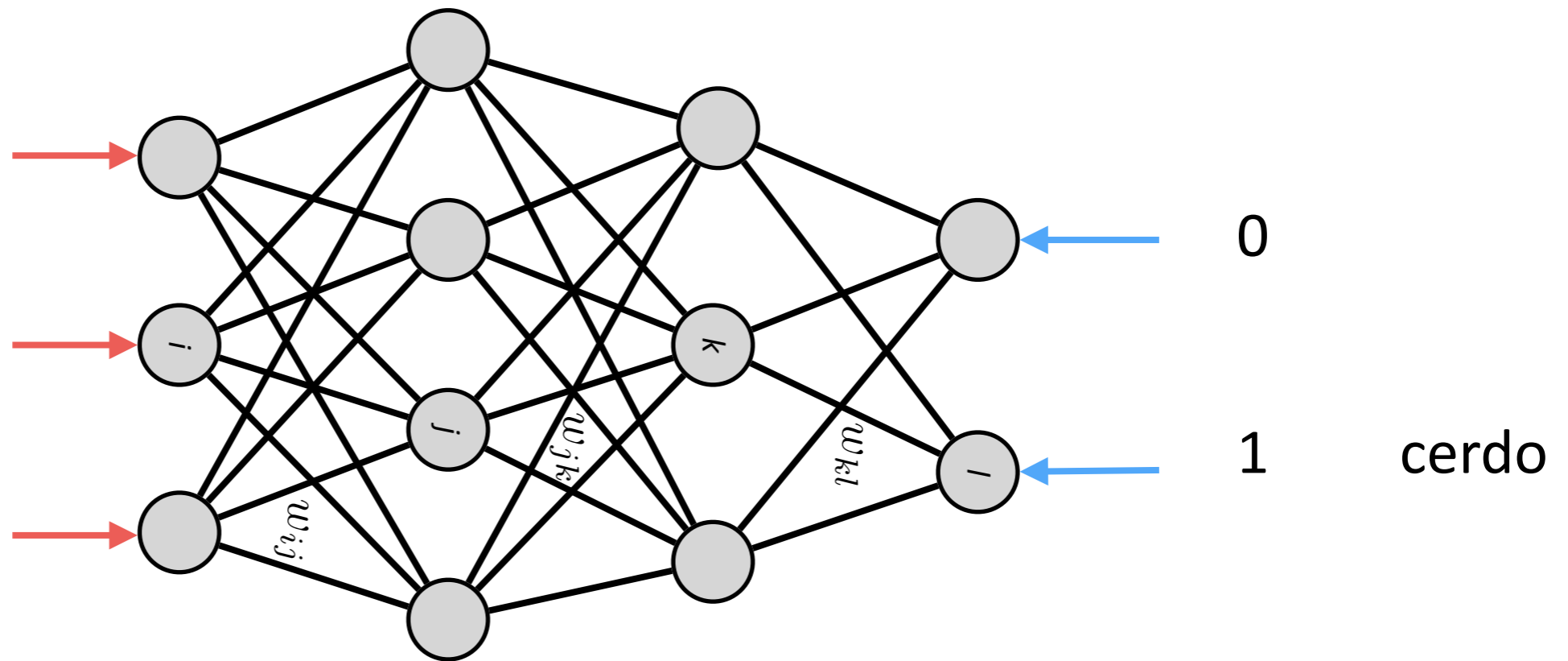
buscar \mathbf{W} t.q. $\sum_i \|\mathbf{y}_i - F(\mathbf{x}_i, \mathbf{W})\|^2$ sea mínimo

regresión/predicción: $\mathbf{y}_i = \{(5.2, 1.1), (-2.1, 0.2), \dots\}$

Redes neuronales

Aprendizaje a partir de ejemplos

Una RN es capaz de **aprender a partir de ejemplos** ajustando los pesos W para minimizar un error
 → problema de optimización



datos $\{\mathbf{x}_i, \mathbf{y}_i\}$

clasificación: $\mathbf{y}_i = \{(0, 1), (1, 0), \dots\}$

buscar \mathbf{W} t.q. $\sum_i \|\mathbf{y}_i - F(\mathbf{x}_i, \mathbf{W})\|^2$ sea mínimo

regresión/predicción: $\mathbf{y}_i = \{(5.2, 1.1), (-2.1, 0.2), \dots\}$

Backpropagation

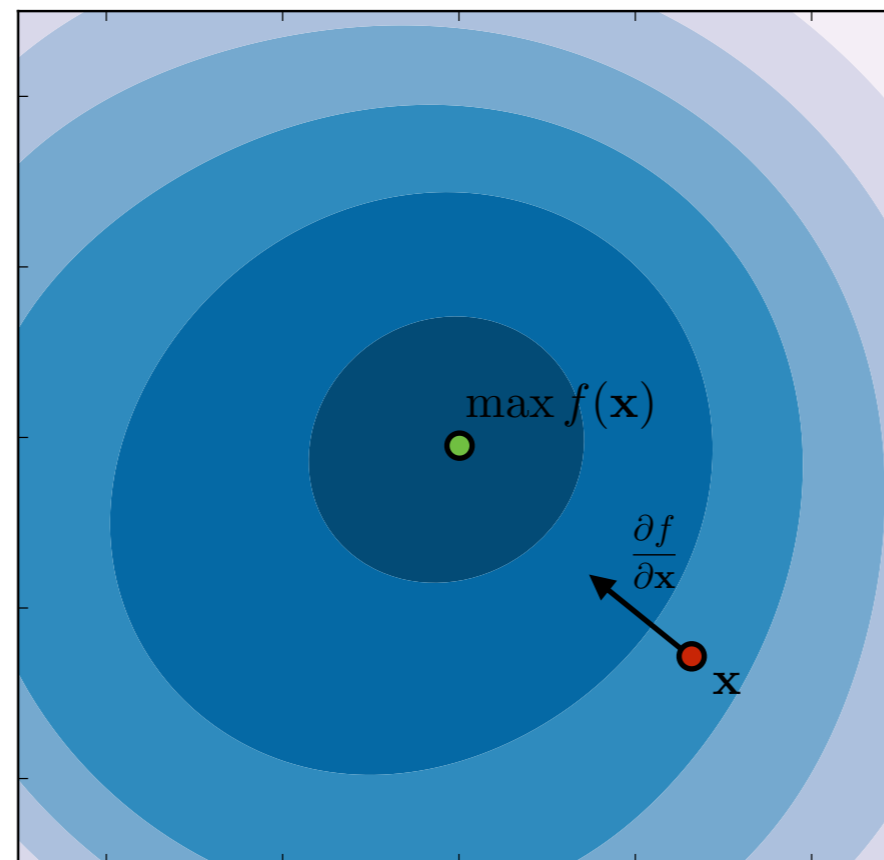
aprendizaje por “retropropagación” del error

El gradiente de una función $f()$
de varias variables

$$\nabla f(x_1, x_2, \dots, x_n) = \frac{\partial f}{\partial \mathbf{x}}$$

es un **vector** en la dirección
de máximo crecimiento de f

$f(\mathbf{x})$



Si definimos el error de una RN como

$$E = \sum_i \| \mathbf{y}_i - F(\mathbf{x}_i, \mathbf{W}) \|^2$$

dado que \mathbf{y}_i , \mathbf{x}_i , son conocidos y fijos (son los ejemplos) tenemos que

$$E = E(\mathbf{W})$$

¡El gradiente del error

$$\nabla E = \frac{\partial E}{\partial \mathbf{W}}$$

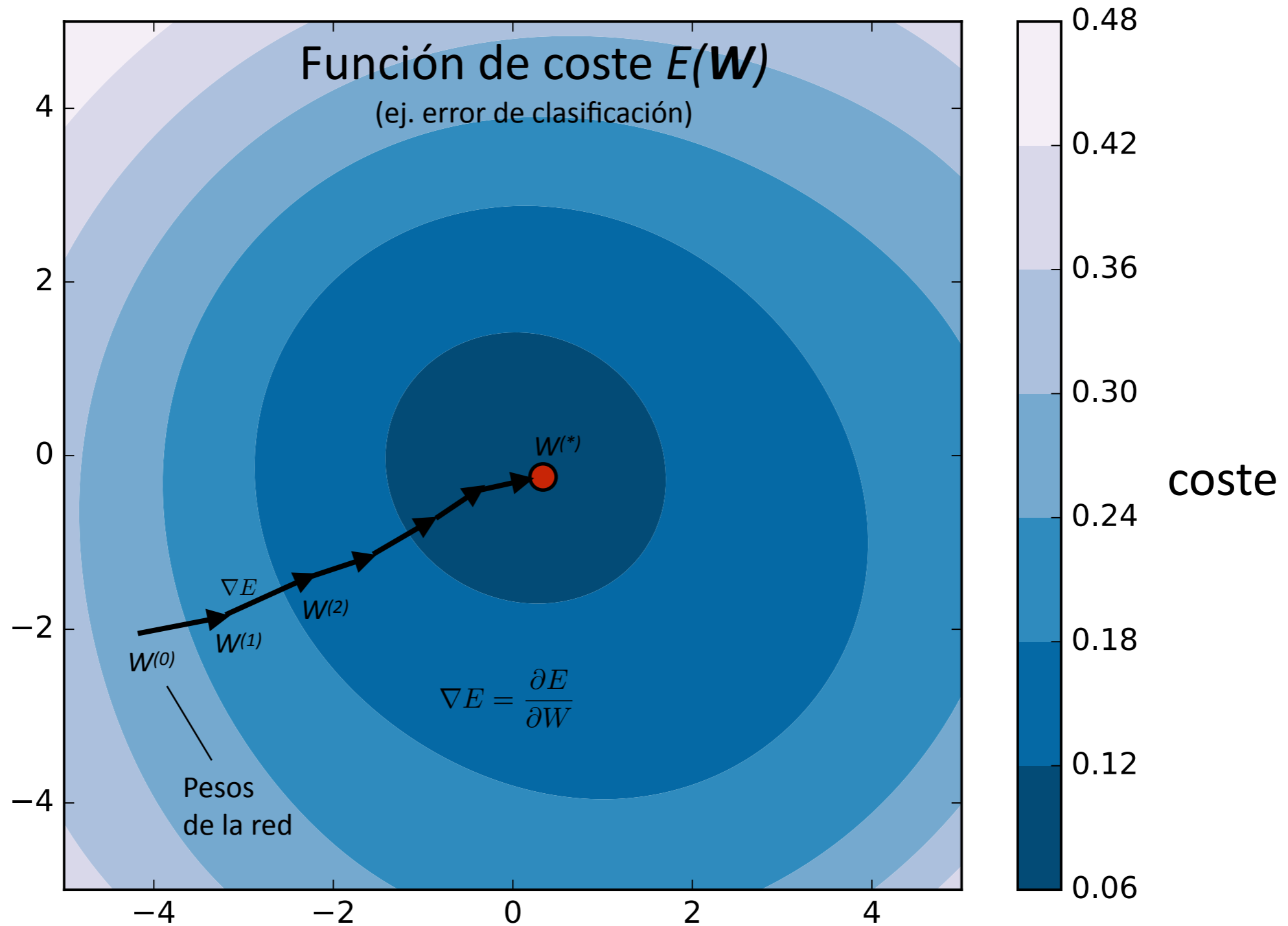
nos dice en qué dirección debemos mover los pesos para disminuir el error!

$$\mathbf{W}^{(k)} \leftarrow \mathbf{W}^{(k-1)} - \mu \frac{\partial E}{\partial \mathbf{W}}$$

**Método
“gradient descent”**

Backpropagation

Descenso del gradiente



Backpropagation

aprendizaje por “retropropagación” del error

Cuestión fundamental del aprendizaje
Determinar el gradiente del error
a partir de los ejemplos

conocidos los ejemplos \mathbf{x}_i , \mathbf{y}_i
y los pesos actuales \mathbf{W}



determinar
el gradiente
que nos dice hacia
dónde mover los pesos

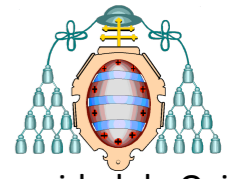
$$\nabla E = \frac{\partial E}{\partial \mathbf{W}}$$

todo indica que va a haber que derivar



Backpropagation

aprendizaje por “retropropagación” del error



Backpropagation es el mecanismo en el que se basa el aprendizaje profundo. Data de los años 80.

Las RN son, básicamente
composiciones de funciones



regla de la cadena
aplicada masivamente



ajusta pesos
en orden inverso a la ejecución
a partir de la diferencia entre la salida real y la deseada

(Rumelhart, Hinton and Williams, 1986)

NATURE VOL. 323 9 OCTOBER 1986 LETTERS TO NATURE 533

Learning representations by back-propagating errors

David E. Rumelhart*, Geoffrey E. Hinton† & Ronald J. Williams*

* Institute for Cognitive Science, C-015, University of California, San Diego, La Jolla, California 92093, USA
† Department of Computer Science, Carnegie-Mellon University, Pittsburgh, Philadelphia 15213, USA

We describe a new learning procedure, back-propagation, for networks of neurone-like units. The procedure repeatedly adjusts the weights of the connections in the network so as to minimize a measure of the difference between the actual output vector of the net and the desired output vector. As a result of the weight adjustments, internal 'hidden' units which are not part of the input or output come to represent important features of the task domain, and the regularities in the task are captured by the interactions of these units. The ability to create useful new features distinguishes back-propagation from earlier, simpler methods such as the perceptron-convergence procedure¹.

There have been many attempts to design self-organizing neural networks. The aim is to find a powerful synaptic modification rule that will allow an arbitrarily connected neural network to develop an internal structure that is appropriate for a particular task domain. The task is specified by giving the desired state vector of the output units for each state vector of the input units. If the input units are directly connected to the output units it is relatively easy to find learning rules that iteratively adjust the relative strengths of the connections so as to progressively reduce the difference between the actual and desired output vectors². Learning becomes more interesting but more difficult when we introduce hidden units whose actual or desired states are not specified by the task. (In perceptrons, there are 'feature analysers' between the input and output that are not true hidden units because their input connections are fixed by hand, so their states are completely determined by the input vector: they do not learn representations.) The learning procedure must decide under what circumstances the hidden units should be active in order to help achieve the desired input-output behaviour. This amounts to deciding what these units should represent. We demonstrate that a general purpose and relatively simple procedure is powerful enough to construct appropriate internal representations.

The simplest form of the learning procedure is for layered networks which have a layer of input units at the bottom; any number of intermediate layers; and a layer of output units at the top. Connections within a layer or from higher to lower layers are forbidden, but connections can skip intermediate layers. An input vector is presented to the network by setting the states of the input units. Then the states of the units in each layer are determined by applying equations (1) and (2) to the connections coming from lower layers. All units within a layer have their states set in parallel, but different layers have their states set sequentially, starting at the bottom and working upwards until the states of the output units are determined.

The total input, x_j , to unit j is a linear function of the outputs, y_i , of the units that are connected to j and of the weights, w_{ji} , on these connections

$$x_j = \sum_i y_i w_{ji} \quad (1)$$

Units can be given biases by introducing an extra input to each unit which always has a value of 1. The weight on this extra input is called the bias and is equivalent to a threshold of the opposite sign. It can be treated just like the other weights.

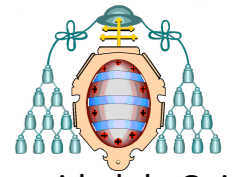
A unit has a real-valued output, y_j , which is a non-linear function of its total input

$$y_j = \frac{1}{1 + e^{-x_j}} \quad (2)$$

† To whom correspondence should be addressed.

Backpropagation

aprendizaje por “retropropagación” del error



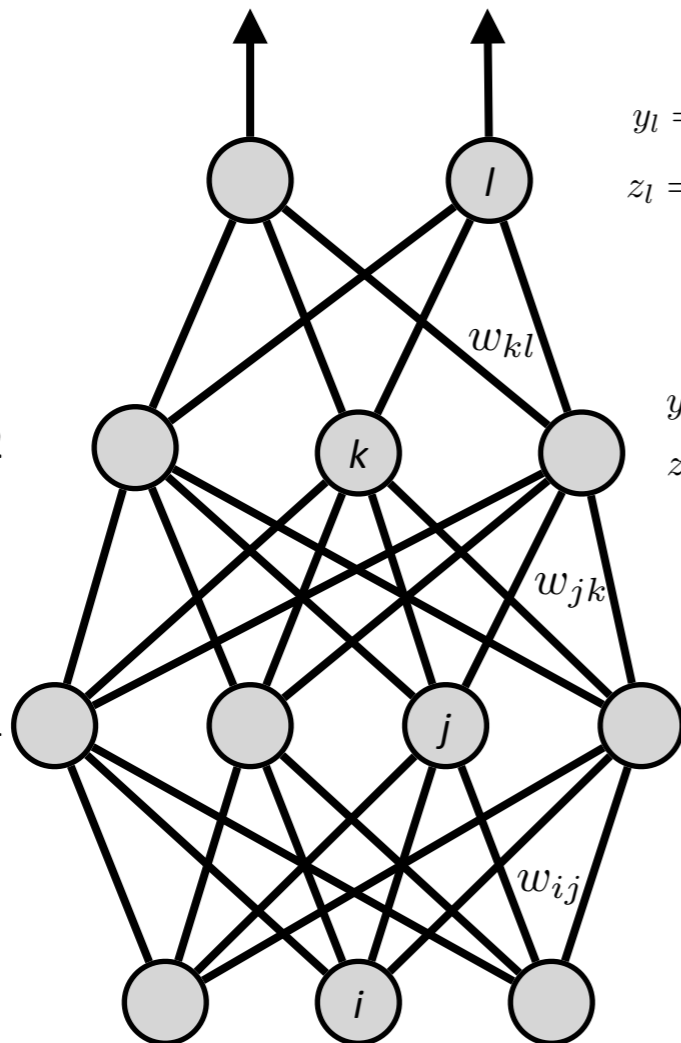
salidas y_l

unidades de salida

capa oculta H2

capa oculta H1

unidades de entrada



entradas x_i

Forward pass

$$y_l = f(z_l)$$

$$z_l = \sum_{k \in H2} w_{kl} y_k$$

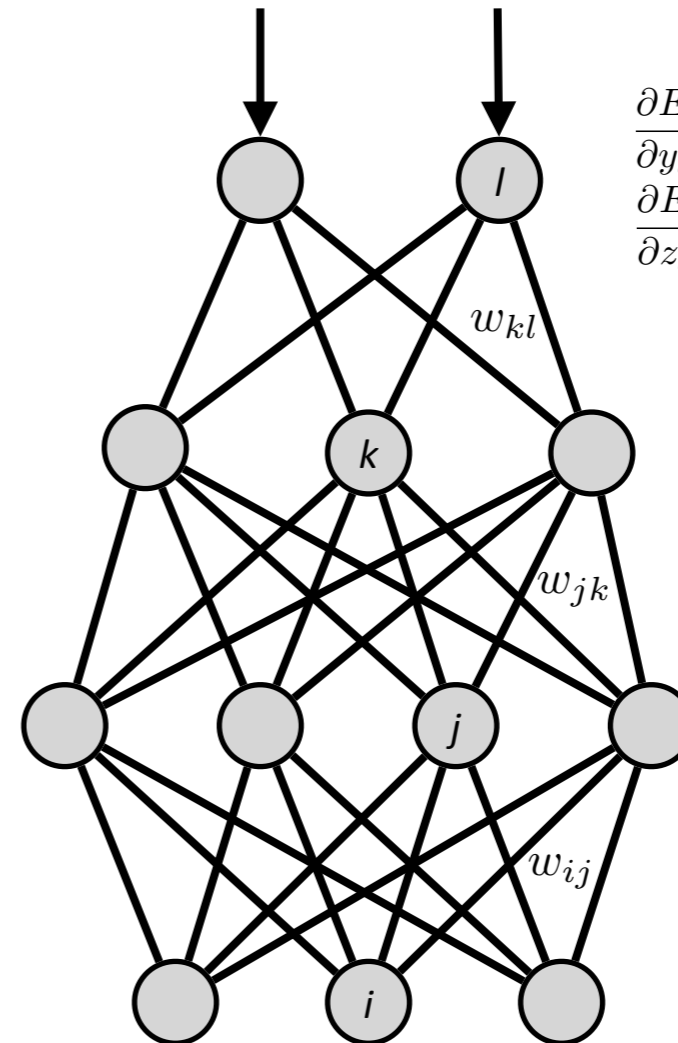
$$y_k = f(z_k)$$

$$z_k = \sum_{j \in H1} w_{jk} y_j$$

$$y_j = f(z_j)$$

$$z_j = \sum_{i \in \text{entrada}} w_{ij} x_i$$

Comparar salidas y_l con objetivos t_l



Backward pass

$$\frac{\partial E}{\partial y_l} = y_l - t_l$$

$$\frac{\partial E}{\partial z_l} = \frac{\partial E}{\partial y_l} \frac{\partial y_l}{\partial z_l} \quad \frac{\partial E}{\partial w_{kl}} = y_k \frac{\partial E}{\partial z_l}$$

$$\frac{\partial E}{\partial y_k} = \sum_{l \in \text{salidas}} w_{kl} \frac{\partial E}{\partial z_l}$$

$$\frac{\partial E}{\partial z_k} = \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial z_k} \quad \frac{\partial E}{\partial w_{jk}} = y_j \frac{\partial E}{\partial z_k}$$

$$\frac{\partial E}{\partial y_j} = \sum_{k \in H2} w_{jk} \frac{\partial E}{\partial z_k}$$

$$\frac{\partial E}{\partial z_j} = \frac{\partial E}{\partial y_j} \frac{\partial y_j}{\partial z_j} \quad \frac{\partial E}{\partial w_{ij}} = x_i \frac{\partial E}{\partial z_j}$$

Backpropagation

aprendizaje por “retropropagación” del error

¿ tengo que hacer todo eso ?

las librerías para deep learning

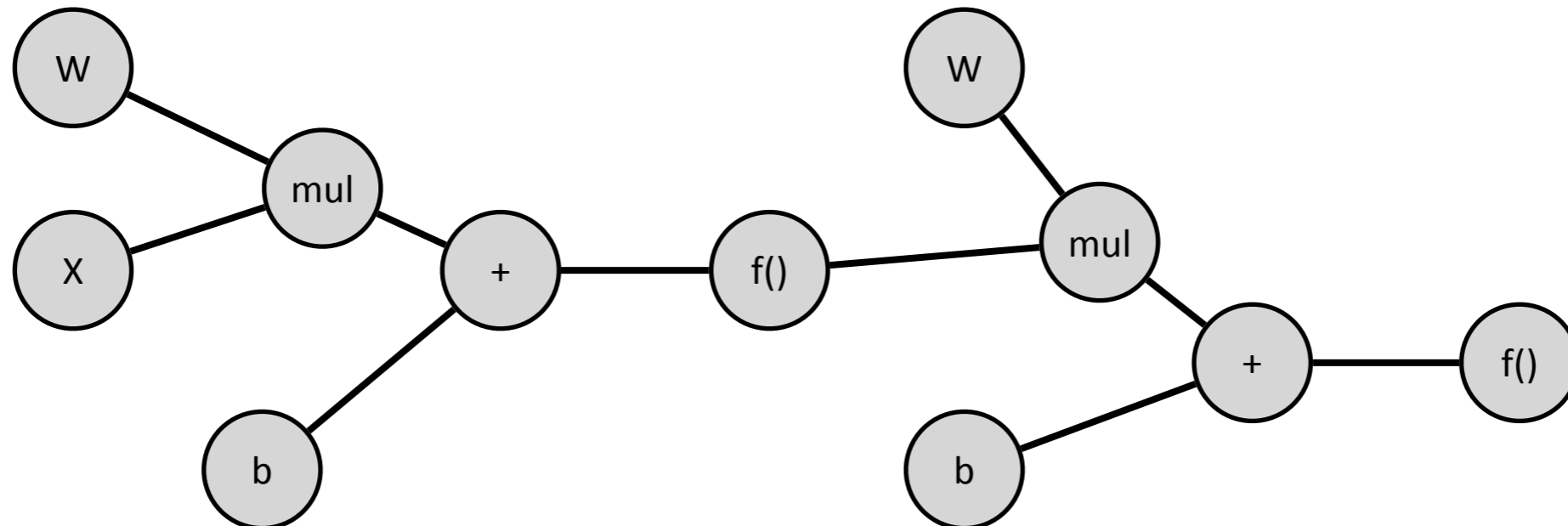


theano

Caffe

DL4J
DEEPLARNING4J

permiten definir la RN como “grafo” de funciones



y calcular el gradiente de los pesos

respecto a una función de coste (“loss function”)

hasta aquí las redes neuronales convencionales...
... pero
¿qué es el deep learning?

Abundancia de **datos**

no necesitamos tanto generalizar

- CIFAR10 (60.000, 32x32, 10 clases)
- imageNet (>14M imágenes etiquetadas)
- Tiny Images

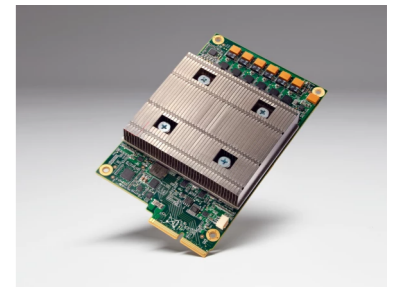
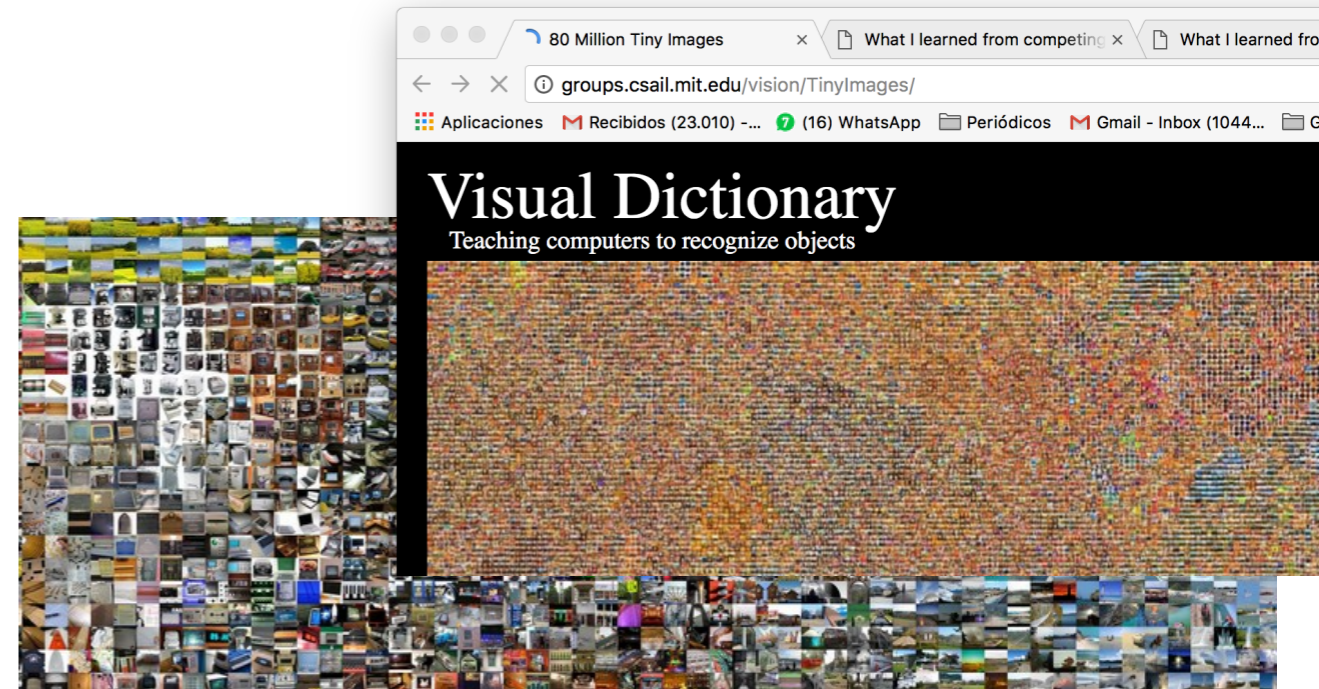
Avances en **computación**

- disponibilidad de CPU's más potentes
- hardware gráfico → GPU's
- hardware específico → TPU's (*tensor processing units*)

Mejoras en los **algoritmos**

- cross-entropy → clasificación
- ReLU (rectifier Linear Units) → vanishing gradient → redes profundas
- convolucionales y pooling → imágenes, audio
- redes recurrentes (LSTM) → secuencias, memoria, lenguaje

Liberación de **librerías de deep learning**



Inteligencia Artificial (IA)

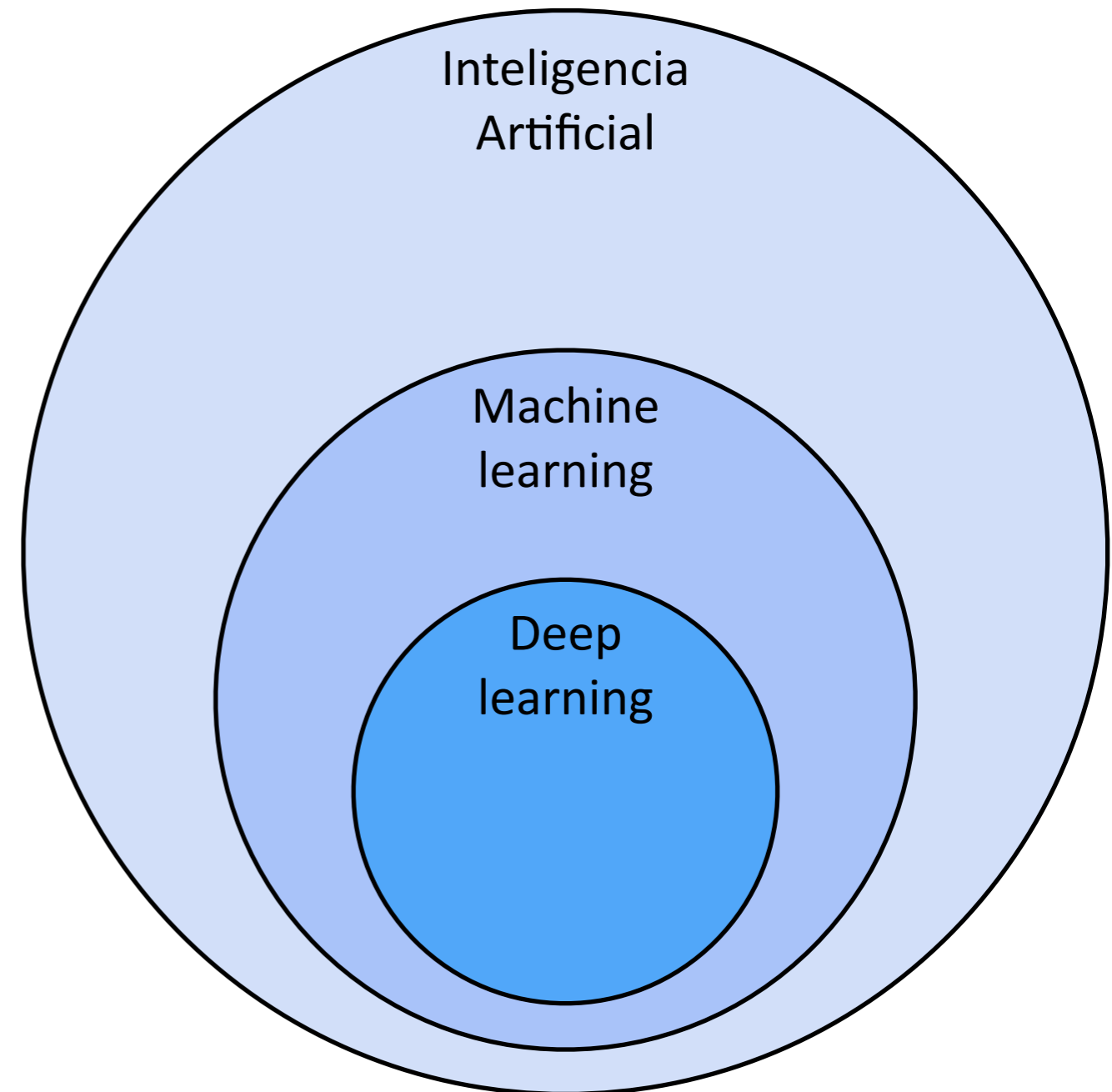
replicación de la inteligencia humana en los computadores

Machine Learning (ML)

(aprendizaje máquina)
habilidad de una máquina para aprender a partir de conjuntos de datos en lugar de a partir de reglas fijas

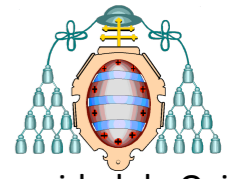
Deep Learning (DL)

(aprendizaje profundo)
implica un aprendizaje a partir de ejemplos utilizando redes neuronales con muchas capas



¿qué caracteriza al deep learning (DL)?

muchas capas, muchas neuronas, muchos pesos ...



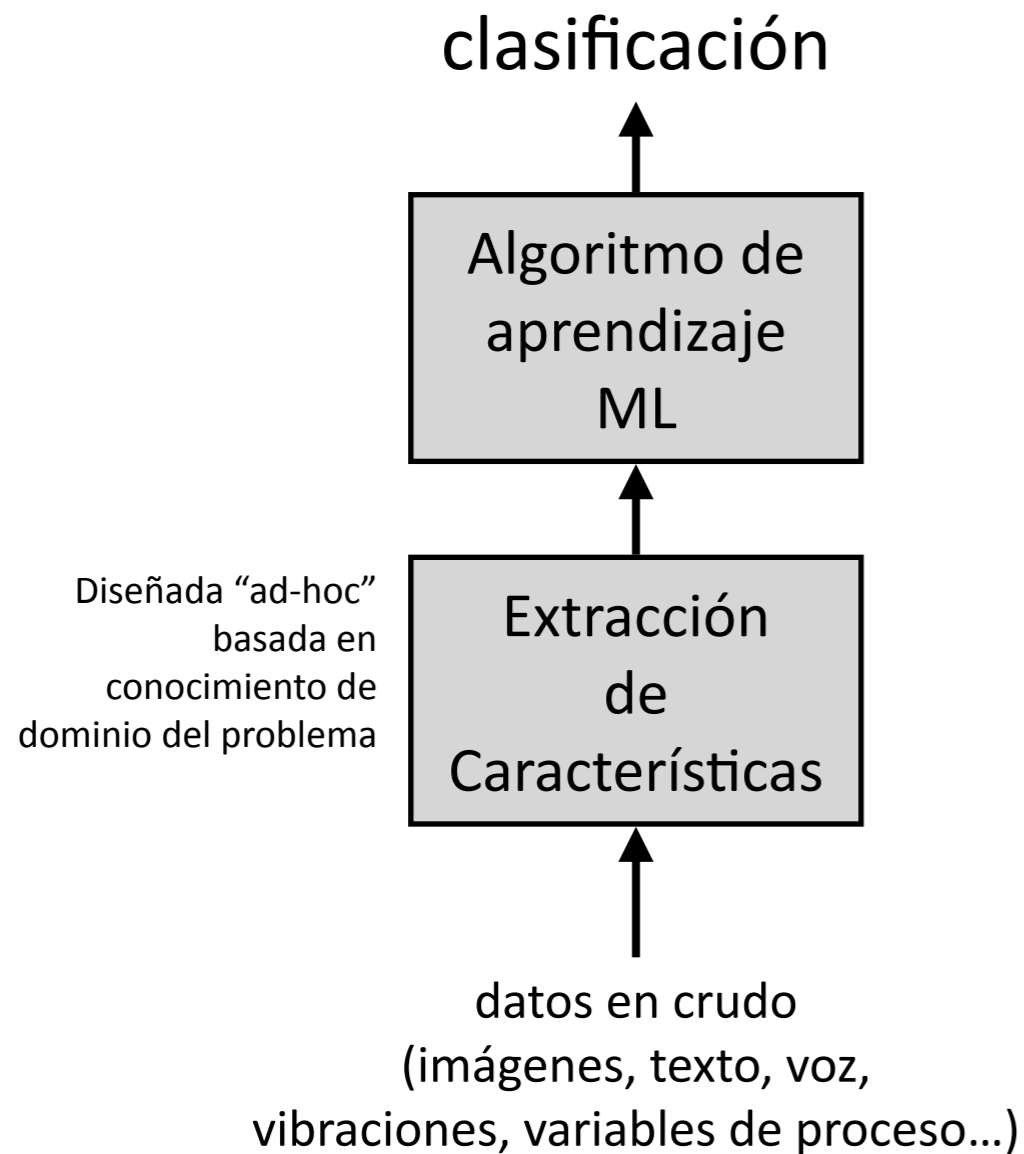
DL utiliza **redes neuronales**
con muchos **niveles** (capas),
organizados **jerárquicamente**

muchas capas
muchas neuronas en cada capa
 $\sim 10^6$ pesos

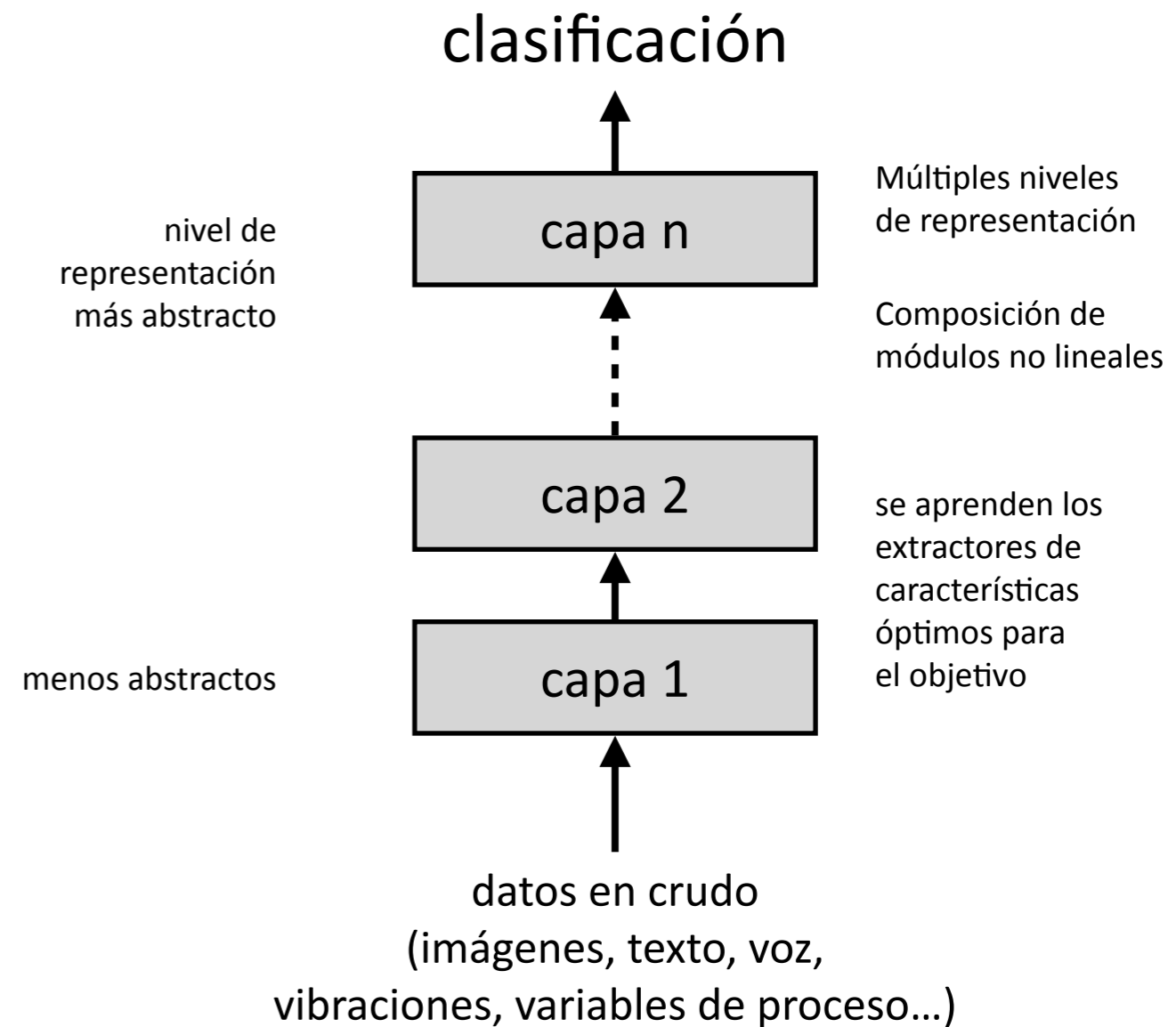
Los primeros niveles detectan
características básicas (bordes, texturas)

Esas características se **componen**
en los niveles superiores,
creando detectores más complejos

Machine Learning convencional

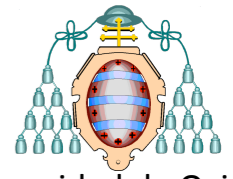


Deep Learning



¿qué caracteriza al deep learning (DL)?

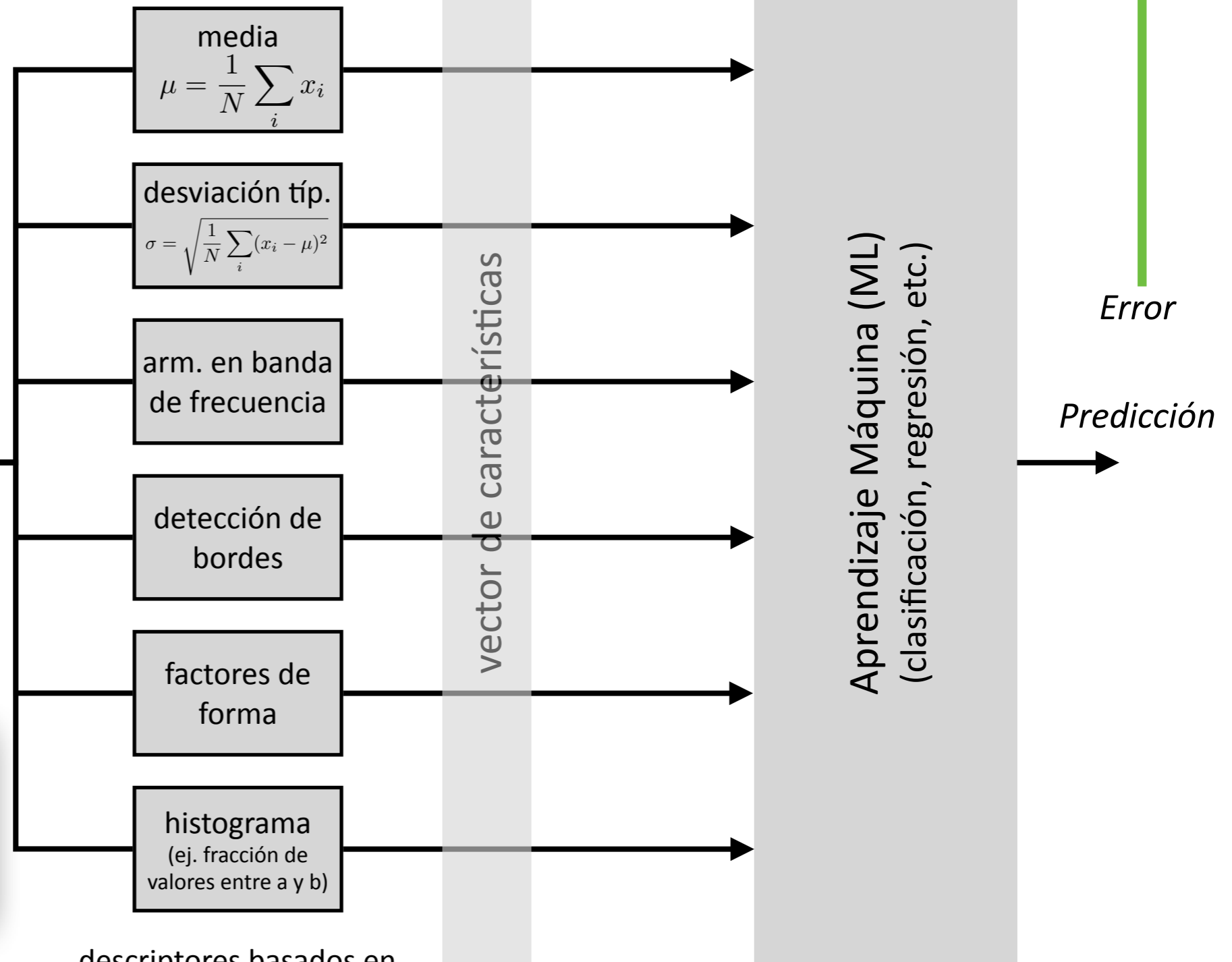
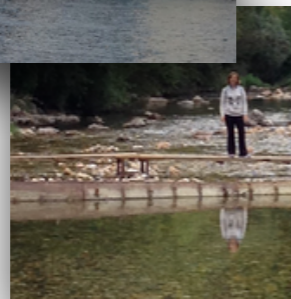
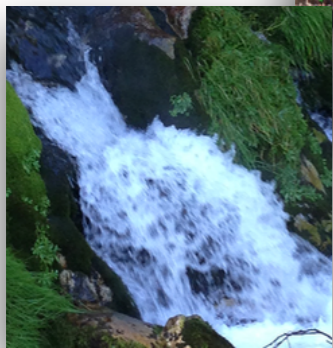
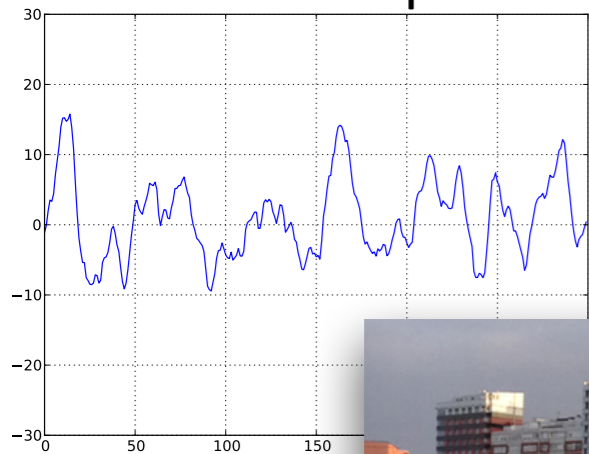
“Representation Learning”



(imágenes,
texto, voz, vibraciones,
variables de proceso...)

datos

“en crudo” del proceso



descriptores basados en
conocimiento de dominio
del problema ⁴⁰

¿qué caracteriza al deep learning (DL)?

“Representation Learning”

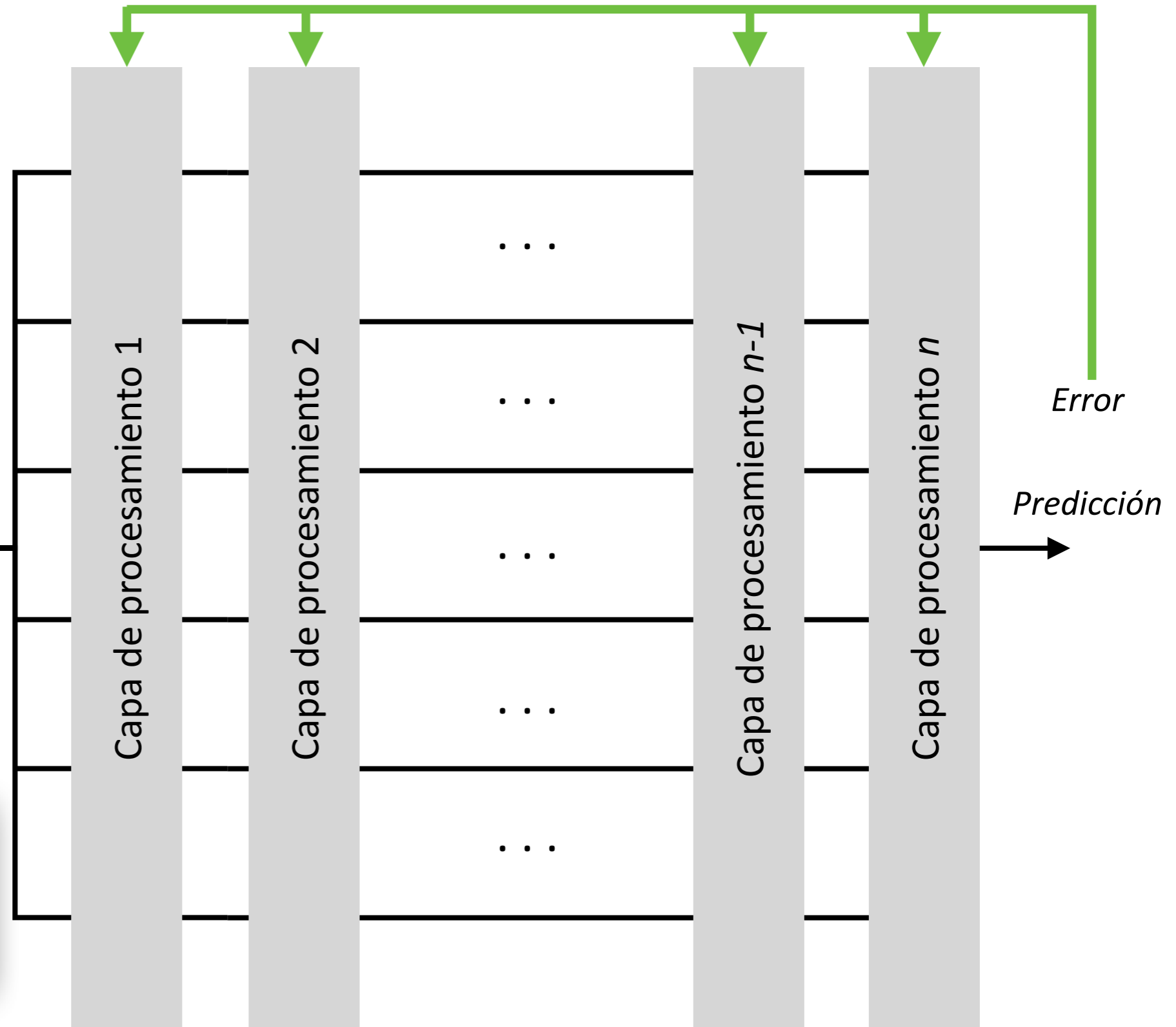
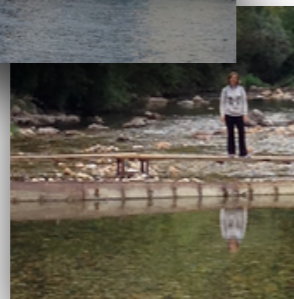
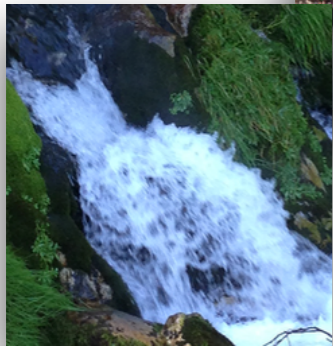
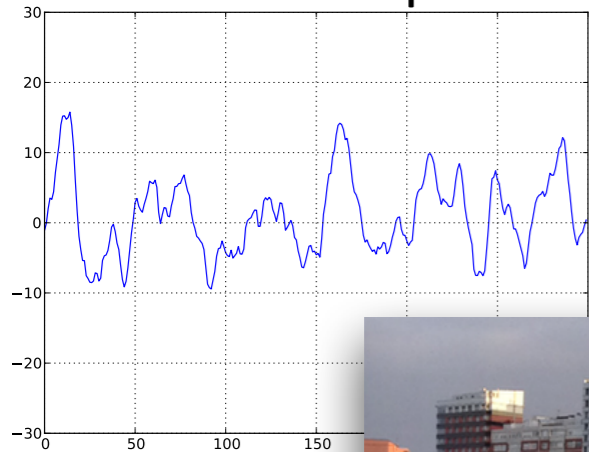
aprendizaje



(imágenes,
texto, voz, vibraciones,
variables de proceso...)

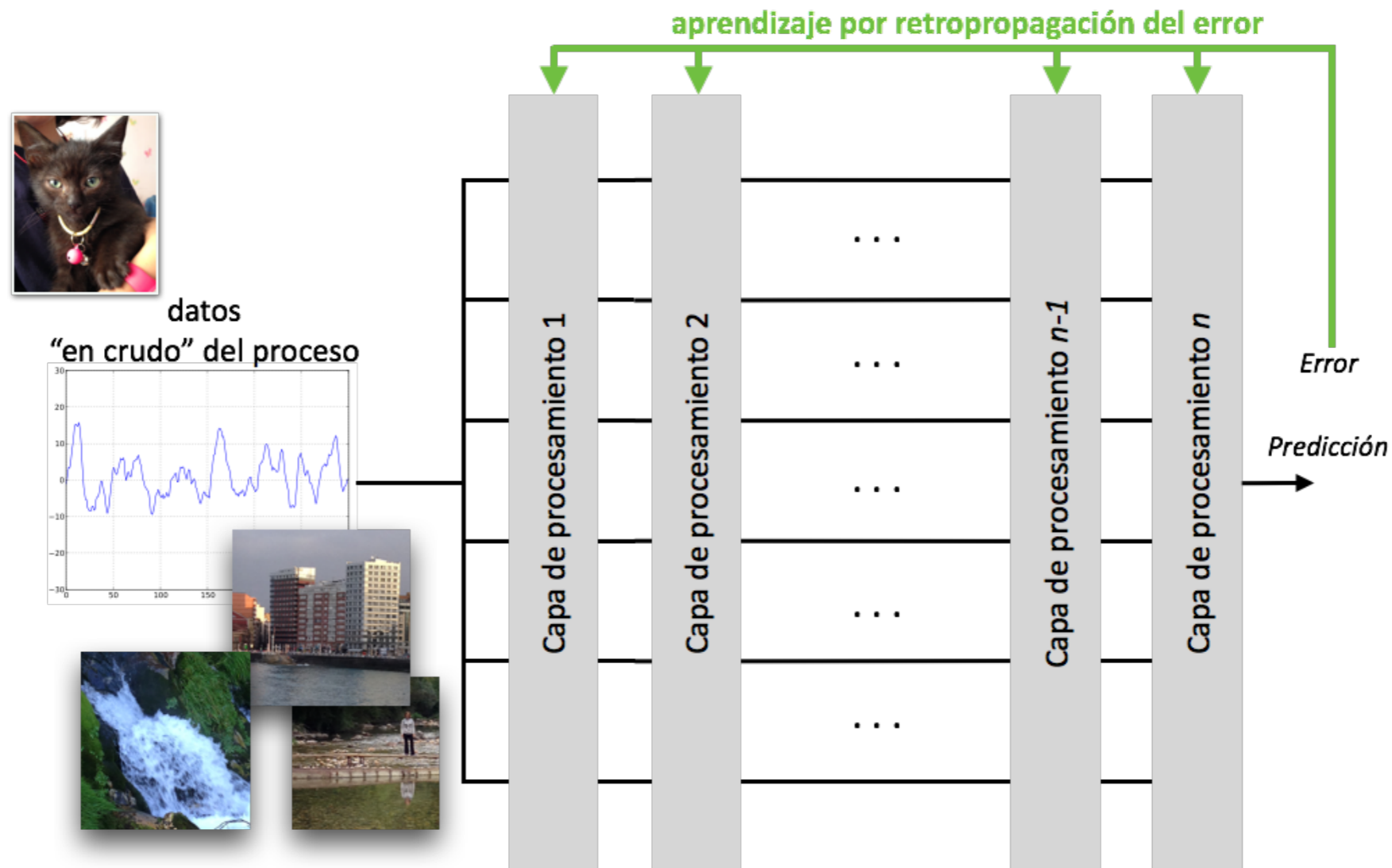
datos

“en crudo” del proceso



¡descriptores aprendidos de ejemplos!

“Representation Learning”



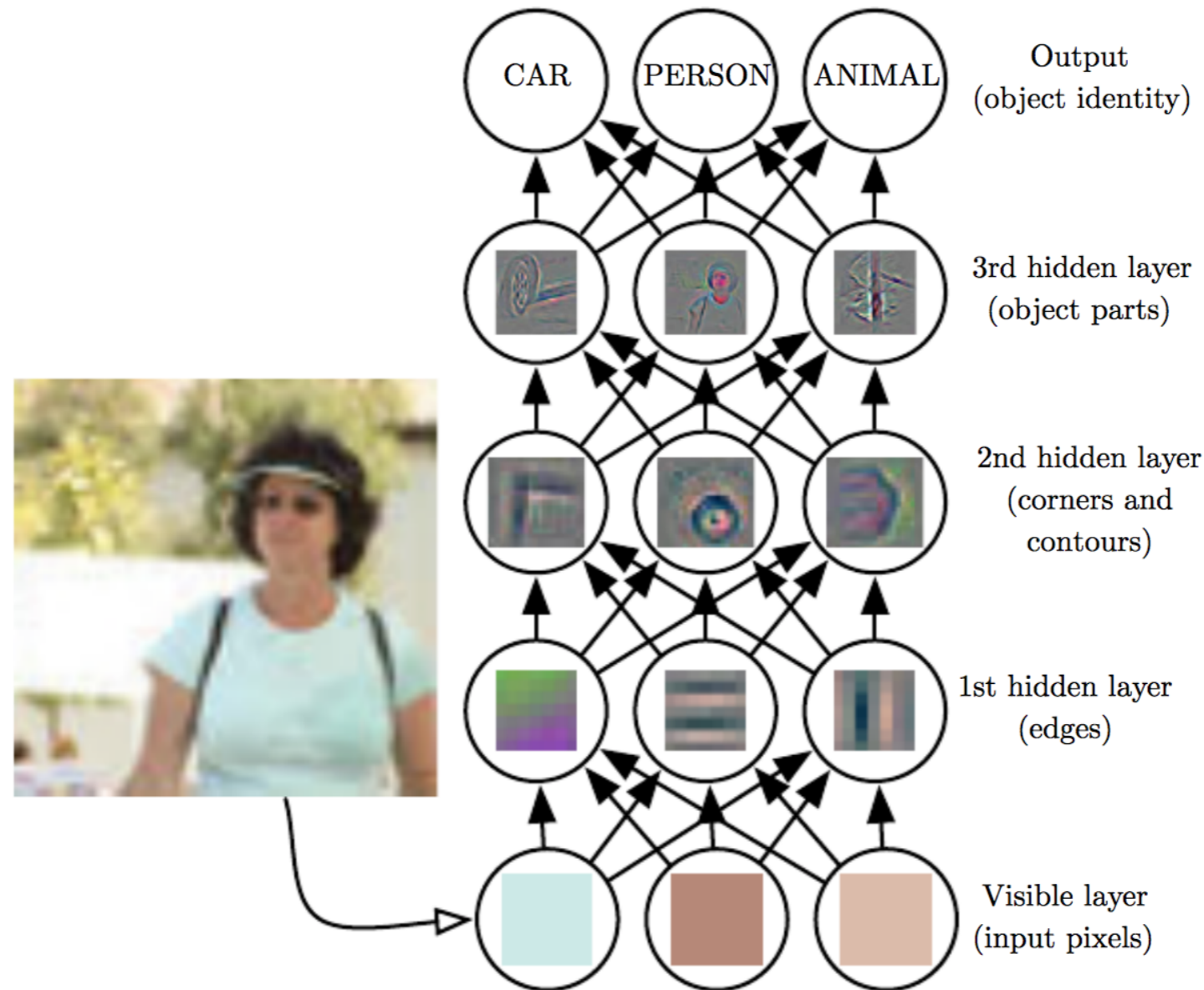
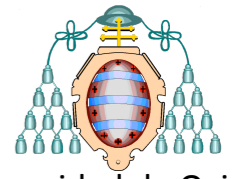
muchas señales naturales
son composiciones de partes más simples
agrupadas de forma jerárquica

**jerarquía
composicional**

bordes → formas → partes → objetos

¿qué caracteriza al deep learning (DL)?

“Representation Learning”



Fuente: I. Goodfellow, Y. Bengio, and A. Courville. Deep learning. MIT Press, 2016.

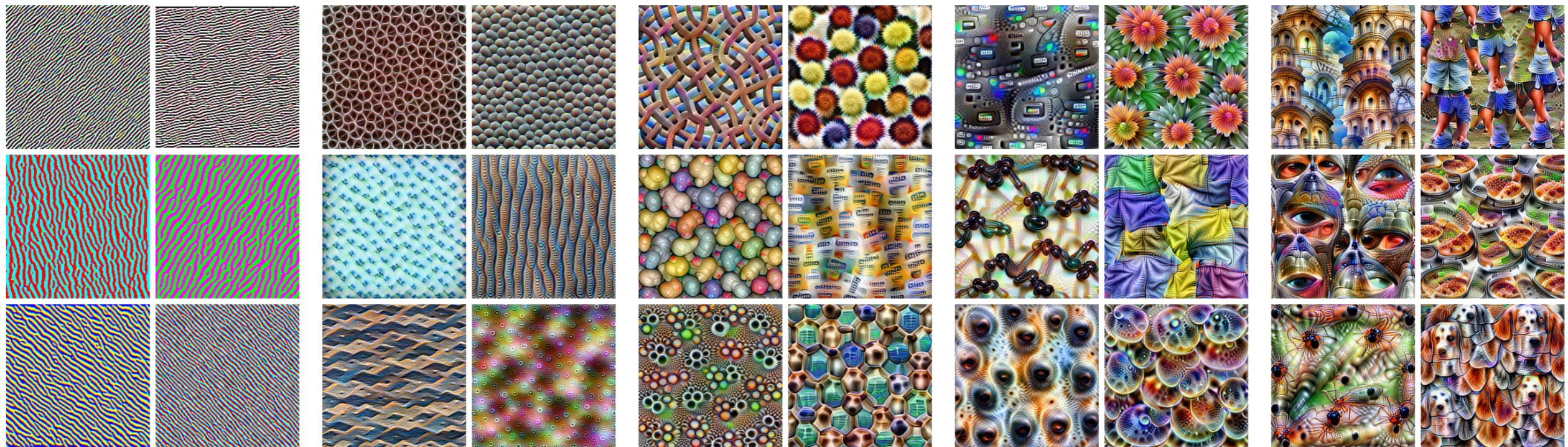
Feature Visualization

Fuente: artículo fantástico en distill.pub

sobre visualización de conocimiento adquirido por redes DL

Chris Olah, Alexander Mordvintsev, Ludwig Schubert → Google Brain / Google Research

<https://distill.pub/2017/feature-visualization/>



Edges (layer conv2d0)

Textures (layer mixed3a)

Patterns (layer mixed4a)

Parts (layers mixed4b & mixed4c)

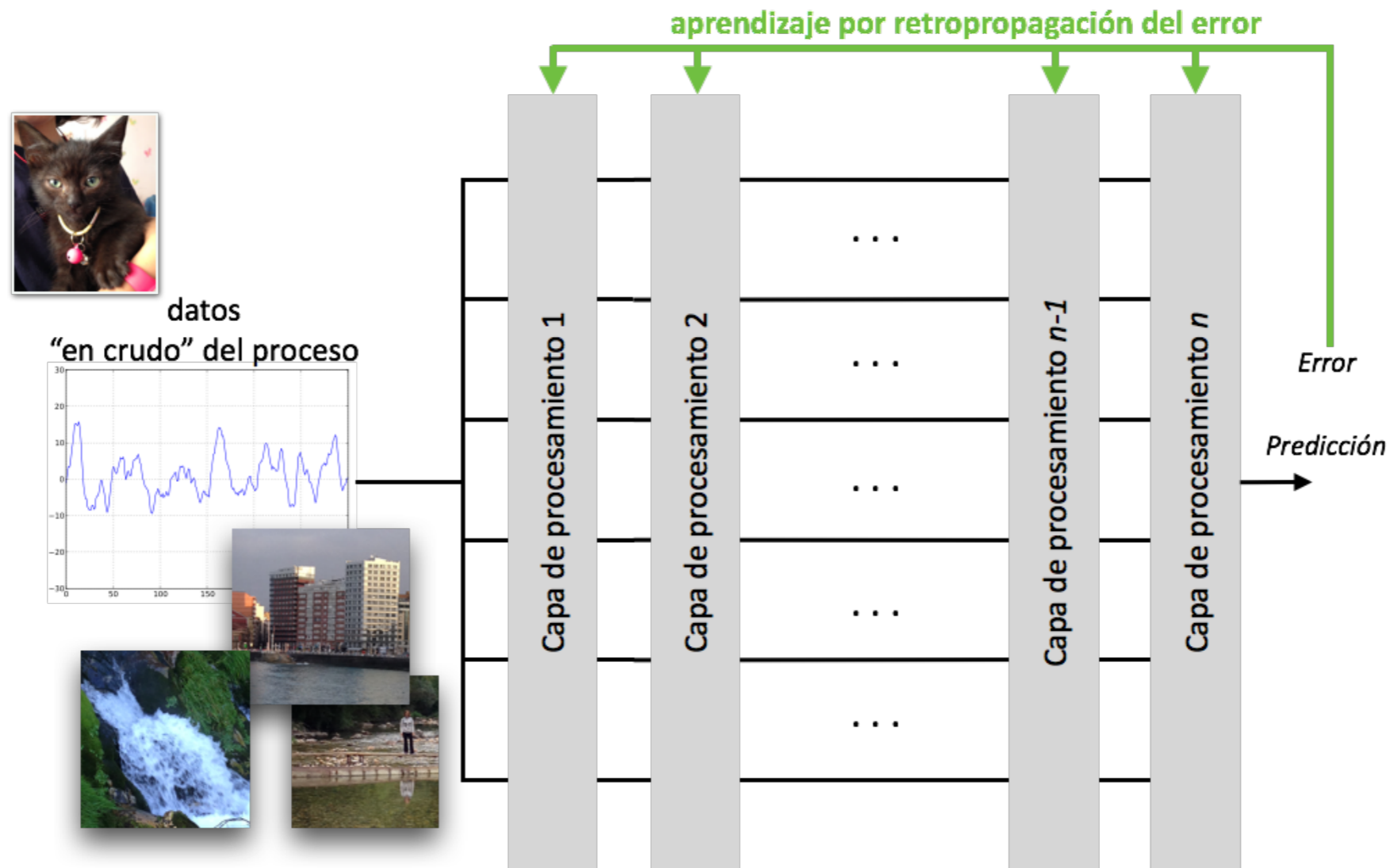
Objects (layers mixed4d & mixed4e)

Red “monstruosa”: GoogleLeNet
 Entrenada en ImageNet (~10⁶ datos)
 Estimar ejemplos que causan cierto

¡Emergen patrones
 de complejidad creciente
 hacia la salida!

¿qué caracteriza al deep learning (DL)?

Complejidad

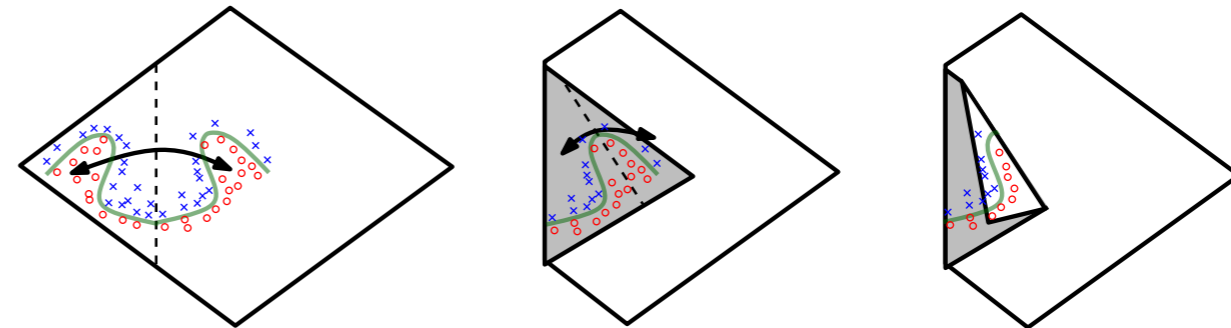


composición de transformaciones \longrightarrow funciones muy complejas

$$f(g(h(j(\dots(x)))))) \longrightarrow (f \circ g \circ h \circ j \dots)(x)$$

Complejidad

Ventaja exponencial



fuente: Montufar et al 2014, "On the Number of Linear Regions of Deep Neural Networks"

Se reutilizan trozos de funciones de forma "composicional"

Una red poco profunda
 requiere exponencialmente más pesos
 que una profunda para aproximar ciertas funciones

Deep Learning es excelente en aprendizaje **no supervisado**, capaz de aprender cosas de ejemplos **sin etiquetar**

capacidad de “desenredar” datos (“disentangling”)

Descubre estructuras complejas en conjuntos de datos enormes

transfer learning

A partir de ahí, puede resolver problemas **supervisados** en los que hay “pocas” etiquetas

¡ Hasta aquí no parece haber muchas diferencias con las RN convencionales !

- mismo entrenamiento → backpropagation
- misma arquitectura

¿lo mismo que antes, pero con más capas?



- **profundidad**

backpropagation “a secas”
no entrena bien capas profundas
(vanishing gradient)

- **generalización**

muchas dimensiones y pocos ejemplos

- imágenes ($10^4 \sim 10^6$ dim)
- texto ($\sim 10^5$ dim)

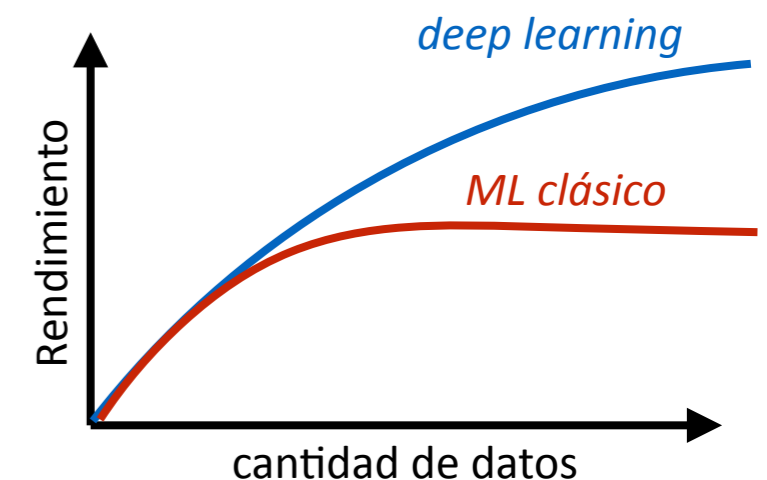
→ generalización

- **complejidad**

optimizar funciones con millones de pesos

- **Escalabilidad**

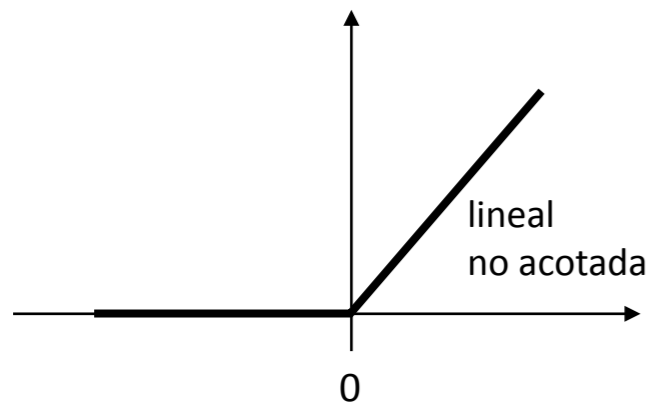
cuantos más datos manejan,
mejor es su comportamiento



Avances clave en DL

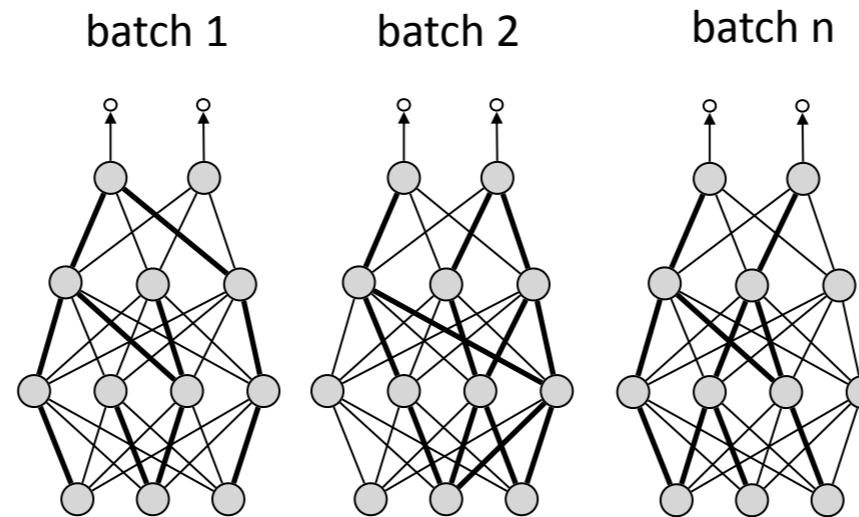
Mejoras en los algoritmos

Funciones de activación ReLu



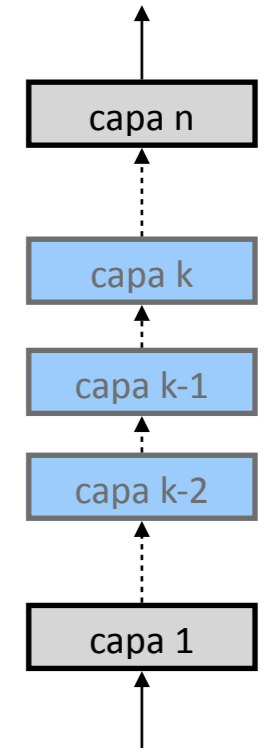
No acotada como la sigmoide
Permite gradientes altos en capas ocultas
Solventa el problema de "shallow gradient"

Regularización (dropout)

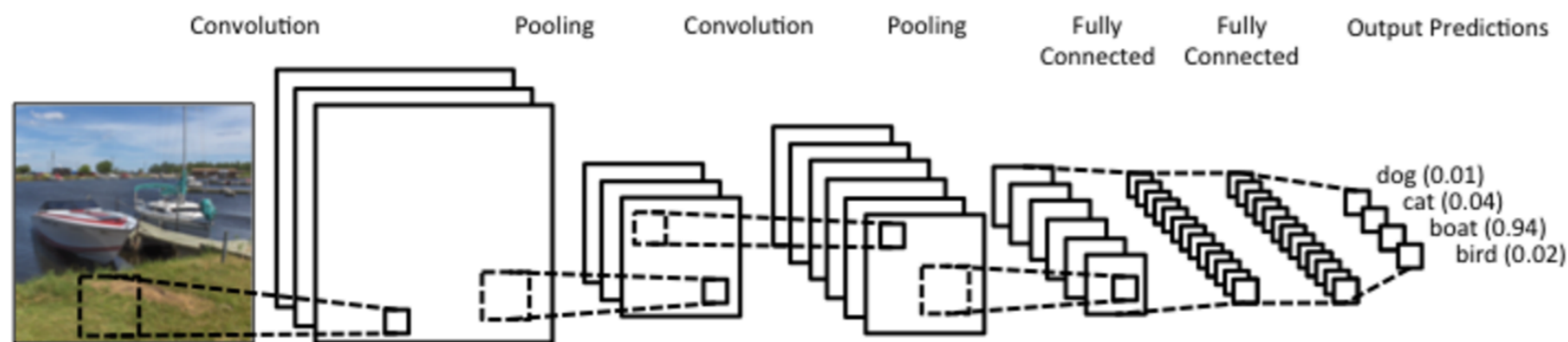


Bloqueo del entrenamiento de algunos pesos,
equivale a un "ensemble" de muchas redes distintas
Ejerce efecto de regularización → mejor generalización

entrenamiento de capas profundas



Redes convolucionales



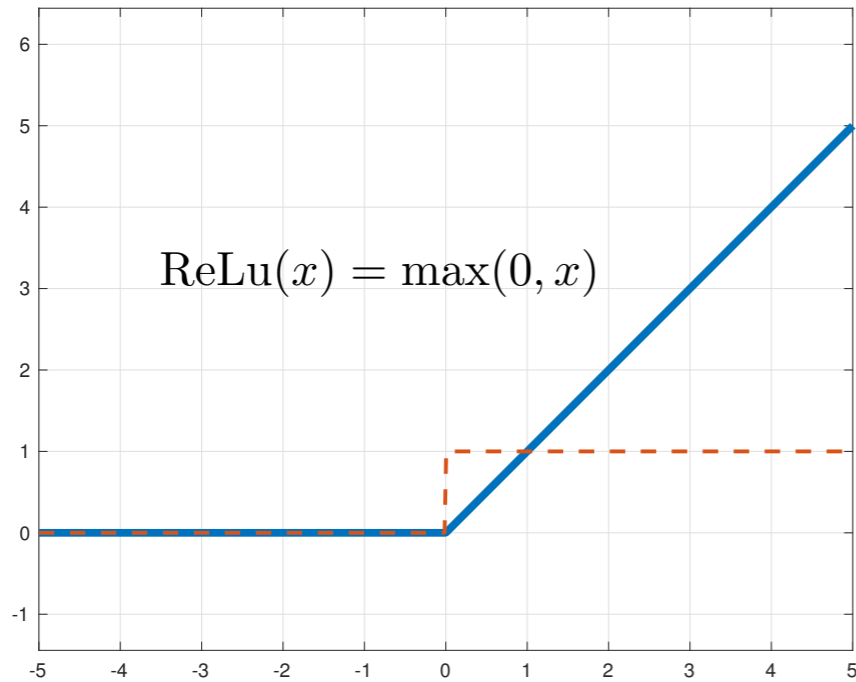
Fuente: <http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/>

op. de convolución → **implementable en GPU**
coherencia → estadísticos locales **invariantes a localización**
(un motivo puede aparecer igual en varios sitios)

Ejemplos sintéticos (rotación, traslación, deformación)
<http://isa.uniovi.es/GSDPI>

Avances clave en DL

Rectifier Linear Units (ReLu)



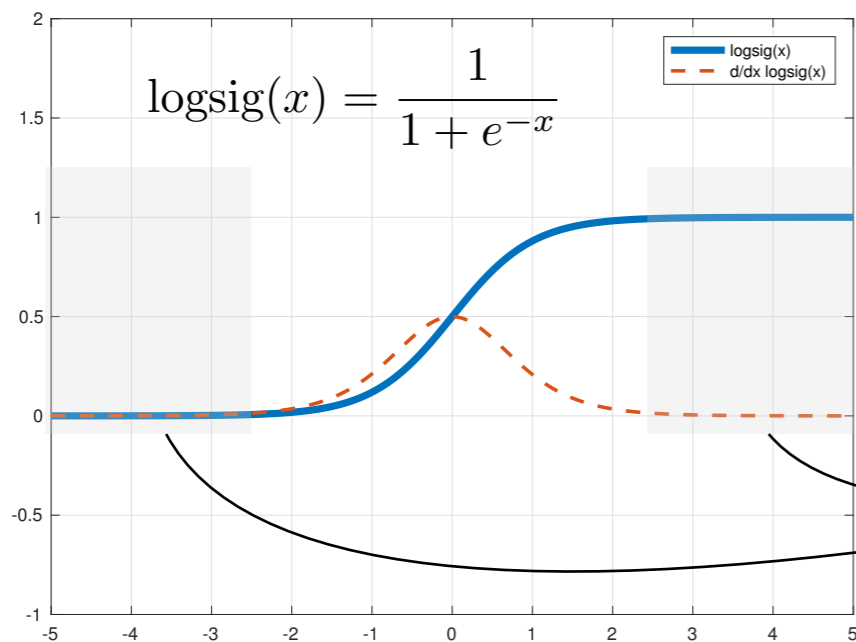
lineal, no acotada



Permite **gradientes altos**
en capas ocultas

Solventa el problema
“shallow gradient”

su caracter “hard” favorece
la dispersión (*sparsity*)



acotada

derivadas casi nulas
en los extremos



gradientes casi nulos
los pesos no avanzan
(la red no aprende)

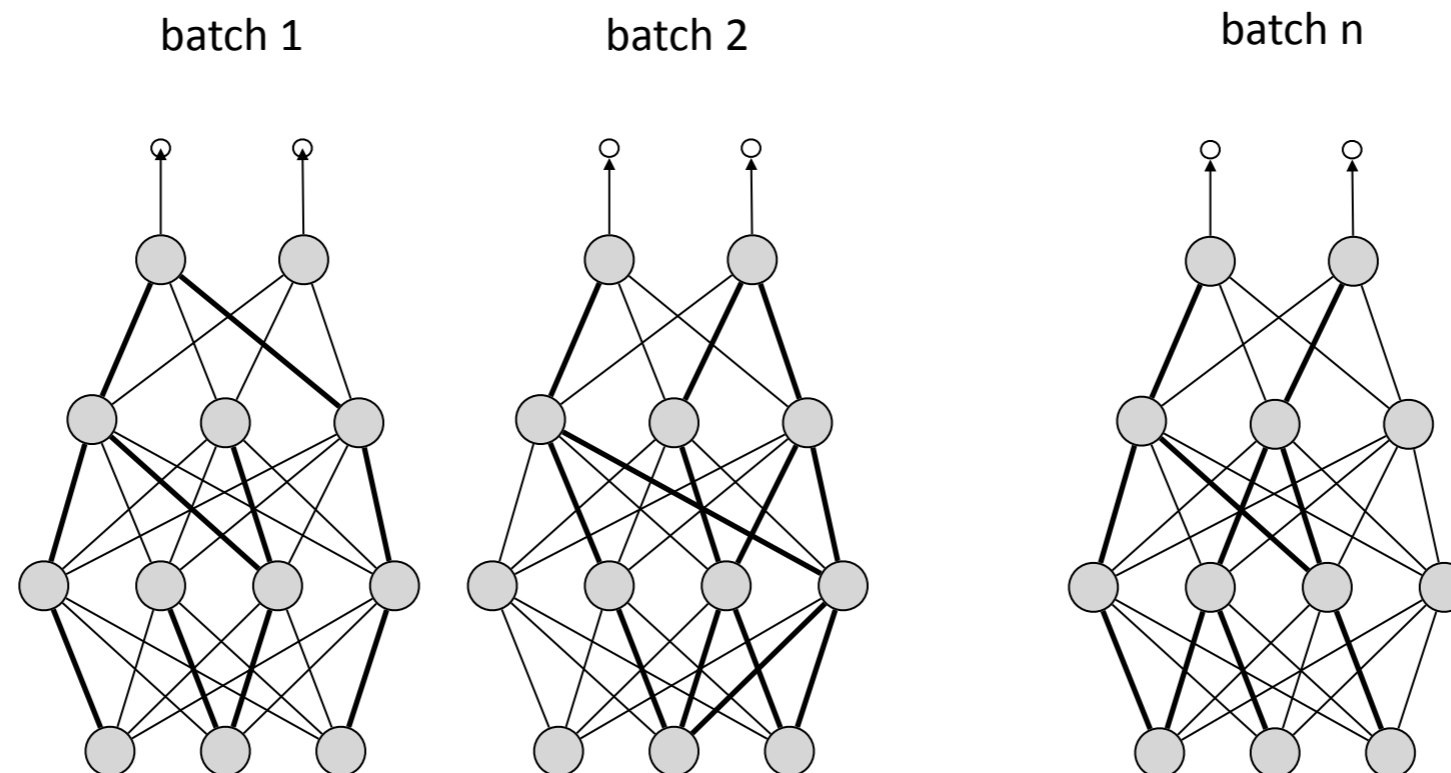
M. D. Zeiler, M. Ranzato, R. Monga, M. Mao, K. Yang, Q. V. Le, P. Nguyen, A. Senior, V. Vanhoucke, J. Dean, et al. On rectified linear units for speech processing. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 3517–3521. IEEE, 2013.

sustituyendo las sigmoidales por ReLu

- se mejora la generalización

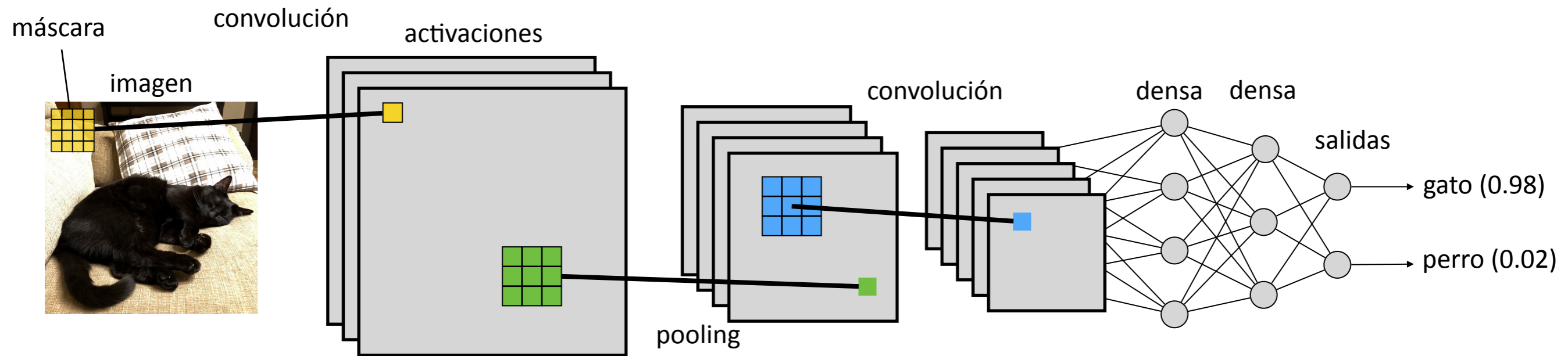
- el aprendizaje DL es más rápido y simple

Bloqueo del entrenamiento de algunos pesos,
equivale a un “ensemble” de muchas redes distintas
Ejerce efecto de regularización → mejor generalización



Avances clave en DL

Redes convolucionales



imagen

5	0	1	0
3	5	1	2
3	2	1	0
4	1	3	1

máscara

1	0	0
0	2	1
0	0	1



activaciones

17	

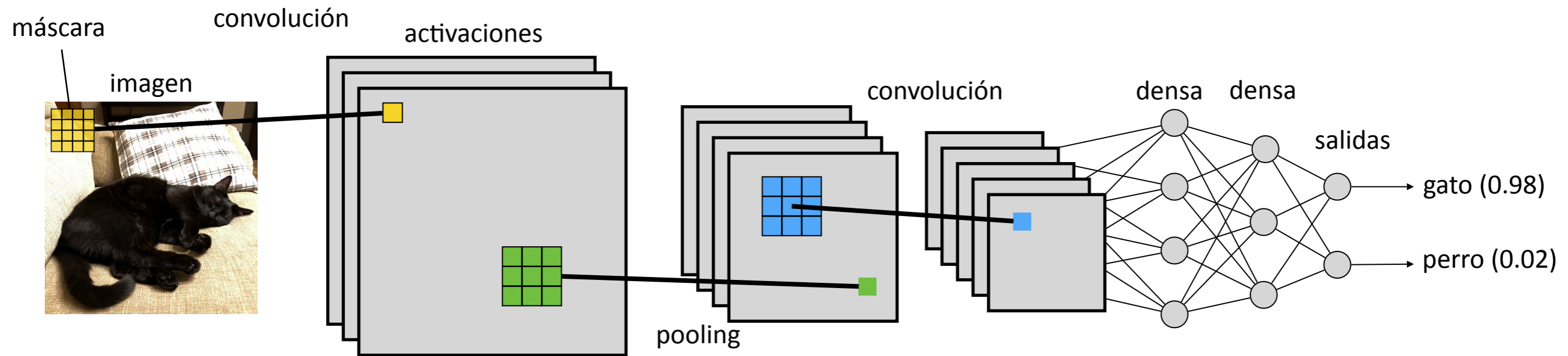
$$5 \times 1 + 0 \times 0 + 1 \times 0 + 3 \times 0 + 5 \times 2 + 1 \times 1 + 3 \times 0 + 2 \times 0 + 1 \times 1 = 17$$

los pesos de la máscara son **adaptados** para alcanzar un objetivo definido por una **función de coste**

una capa convolución usa **muchos menos pesos** que una densa (todos con todos)

Explota principios de **coherencia** (estadísticos locales se preservan) y **jerarquía composicional** los elementos se unen como piezas de un lego

Suelen utilizarse **ejemplos sintéticos** (rotación, traslación, deformación)



original

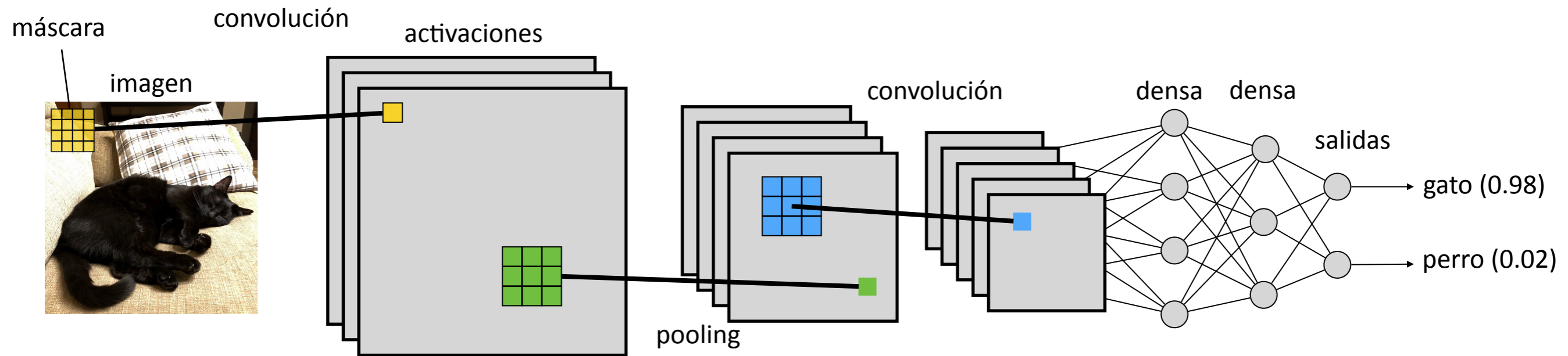
4	0	1	1	1
3	5	1	1	1
3	2	1	1	1
4	1	3	3	3
4	1	3	3	3

máscara 3x3

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

filtrada

Operación de convolución



original

4	0	1	1	1
3	5	1	1	1
3	2	1	1	1
4	1	3	3	3
4	1	3	3	3

máscara 3x3

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

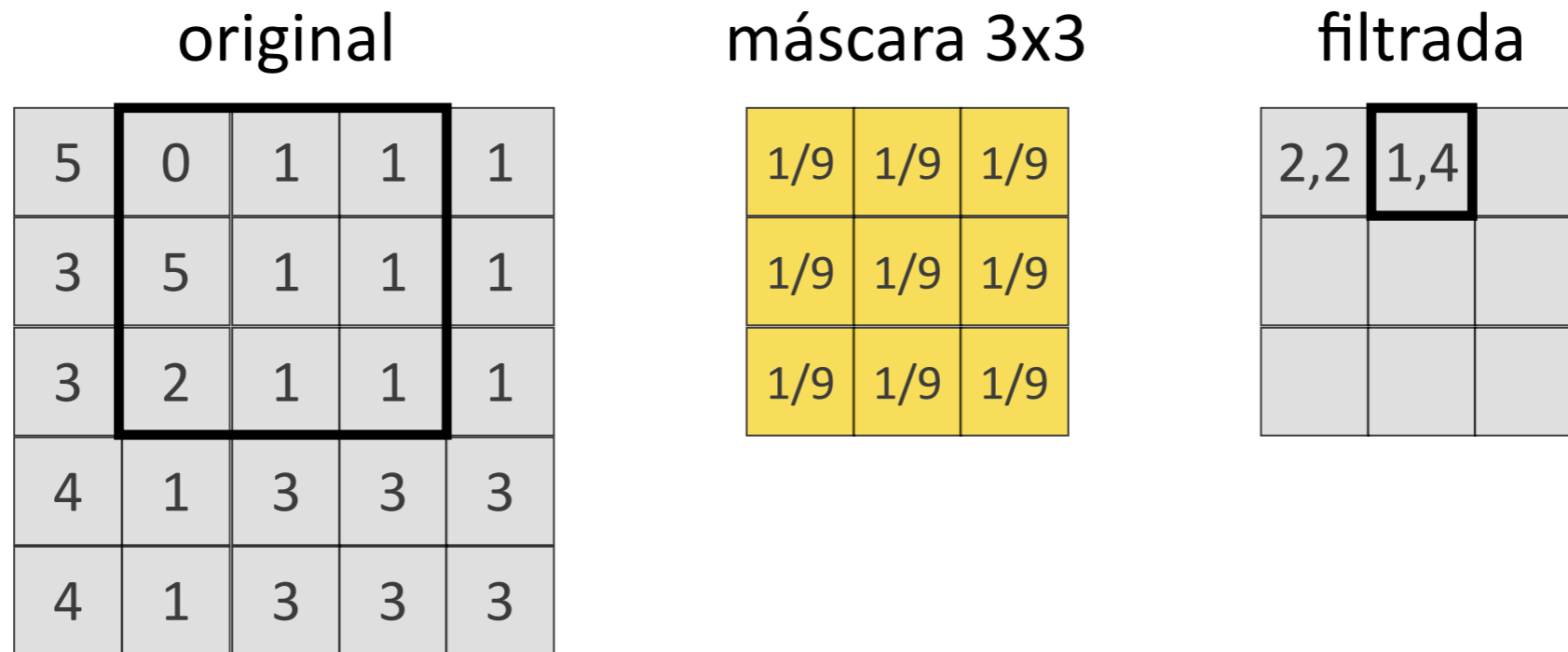
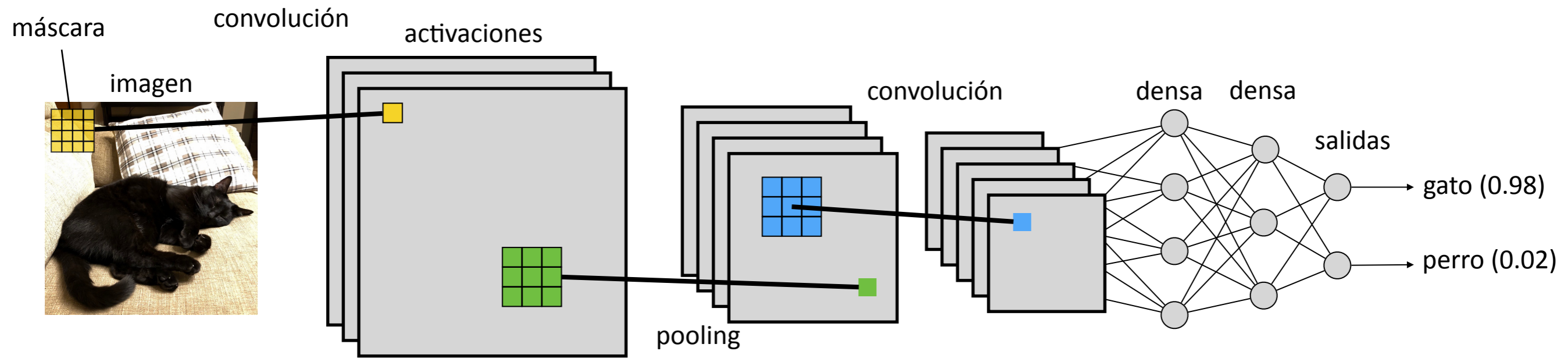
filtrada

2,2		

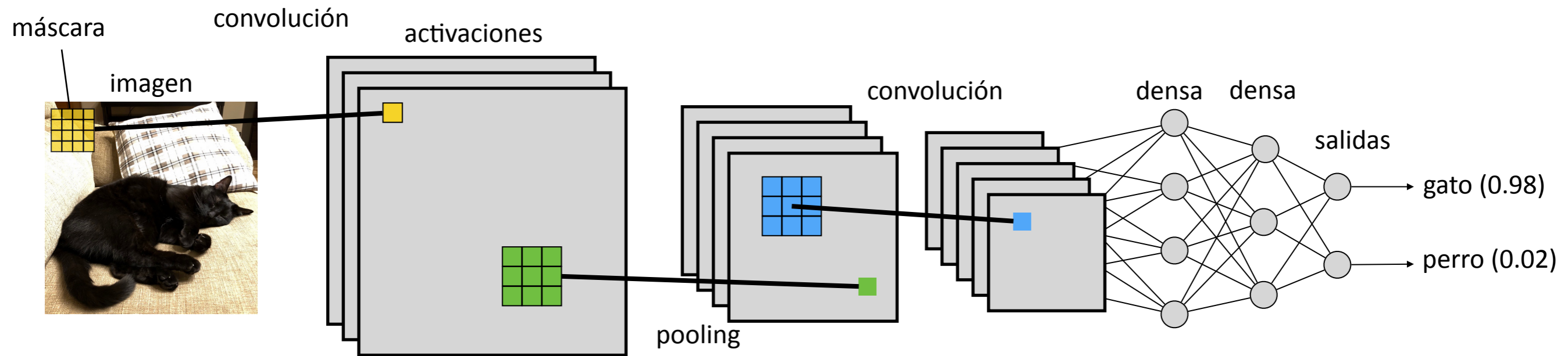
Operación de convolución

Avances clave en DL

Redes convolucionales



Operación de convolución



original

5	0	1	1	1
3	5	1	1	1
3	2	1	1	1
4	1	3	3	3
4	1	3	3	3

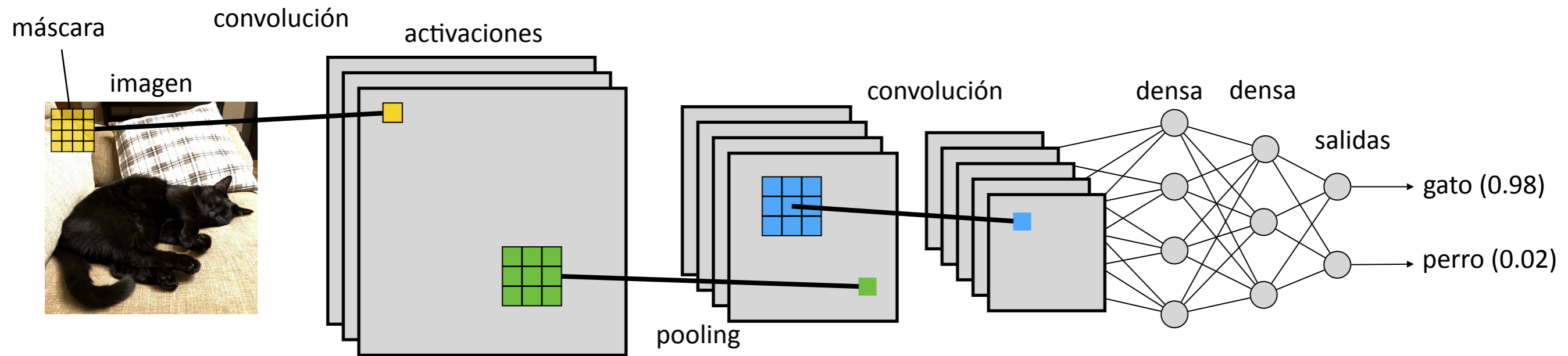
máscara 3x3

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

filtrada

2,2	1,4	1

Operación de convolución



original

5	0	1	1	1
3	5	1	1	1
3	2	1	1	1
4	1	3	3	3
4	1	3	3	3

máscara 3x3

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

filtrada

2,2	1,4	1
2,5		

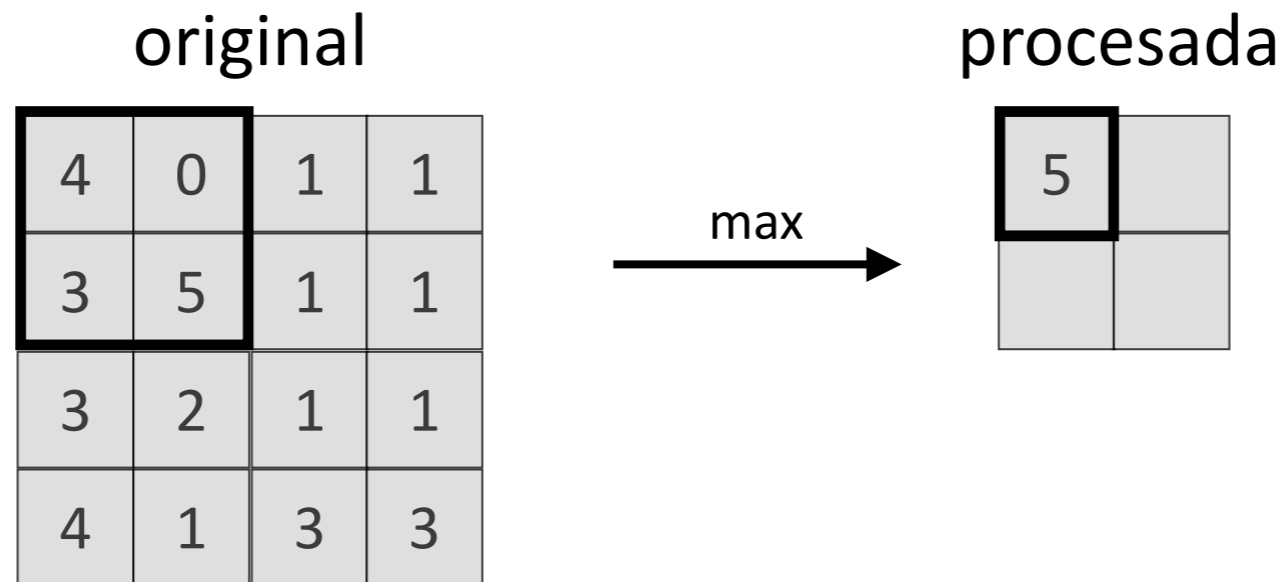
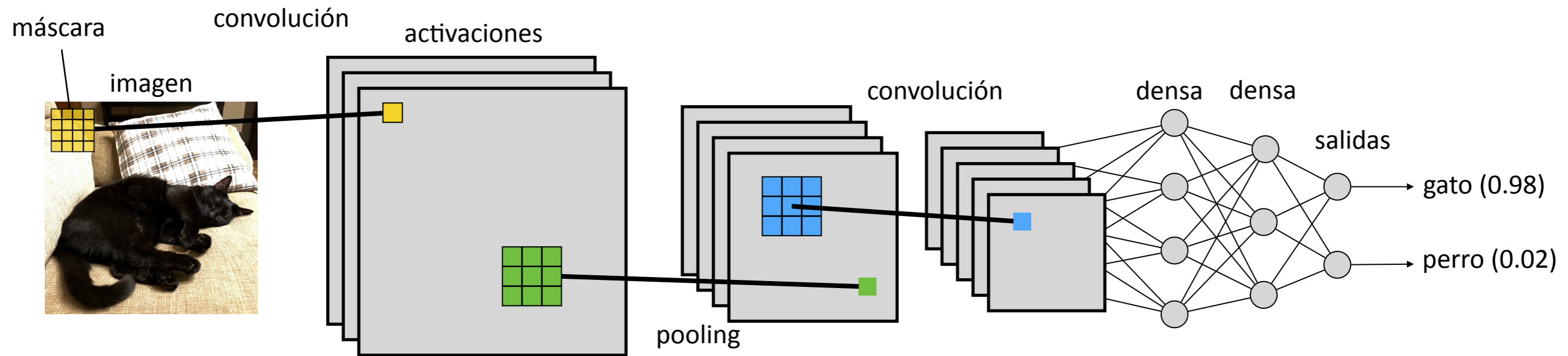
Operación de convolución

La convolución es un filtro

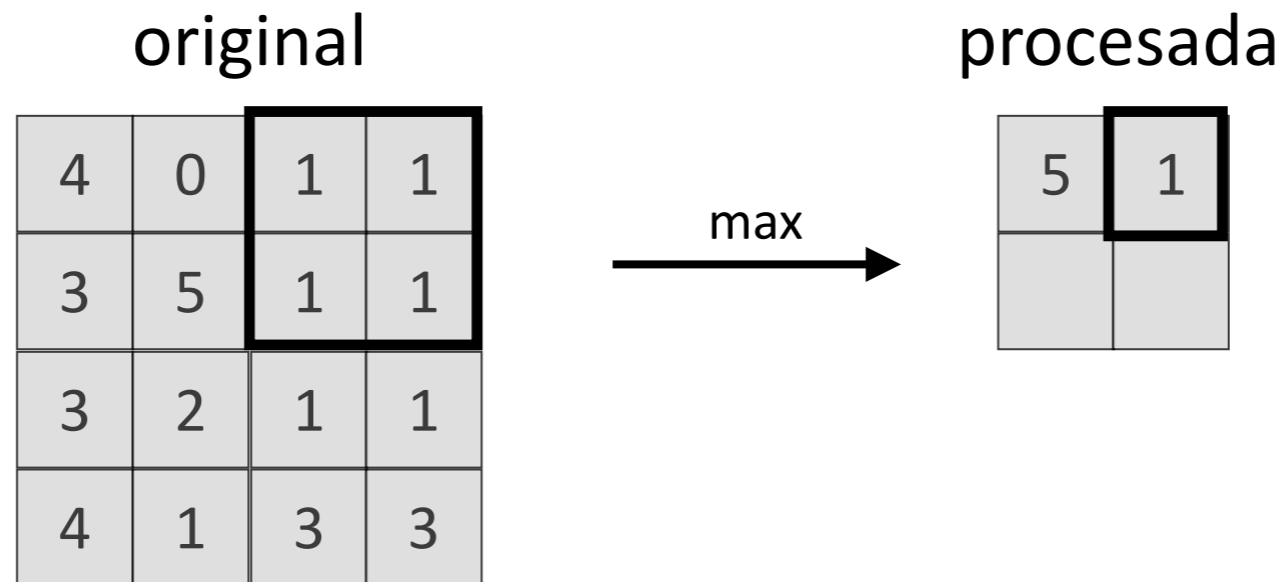
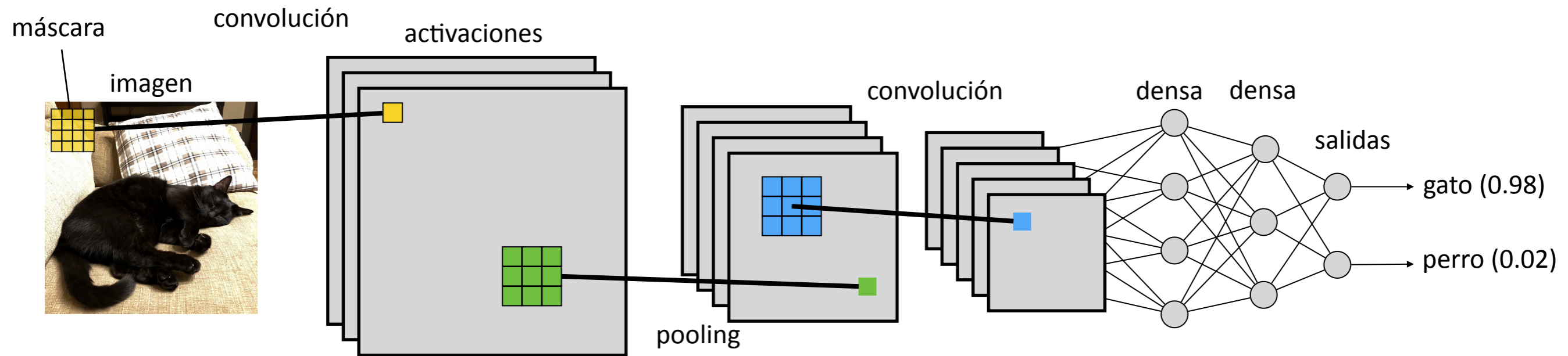
sensible a patrones concretos

depende de la máscara

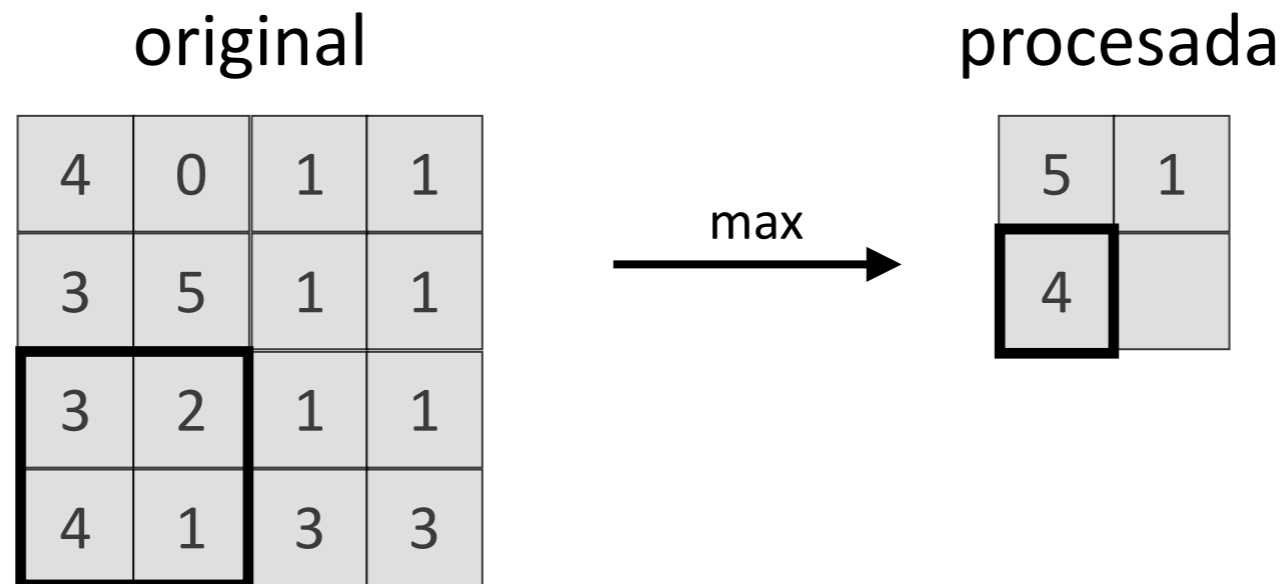
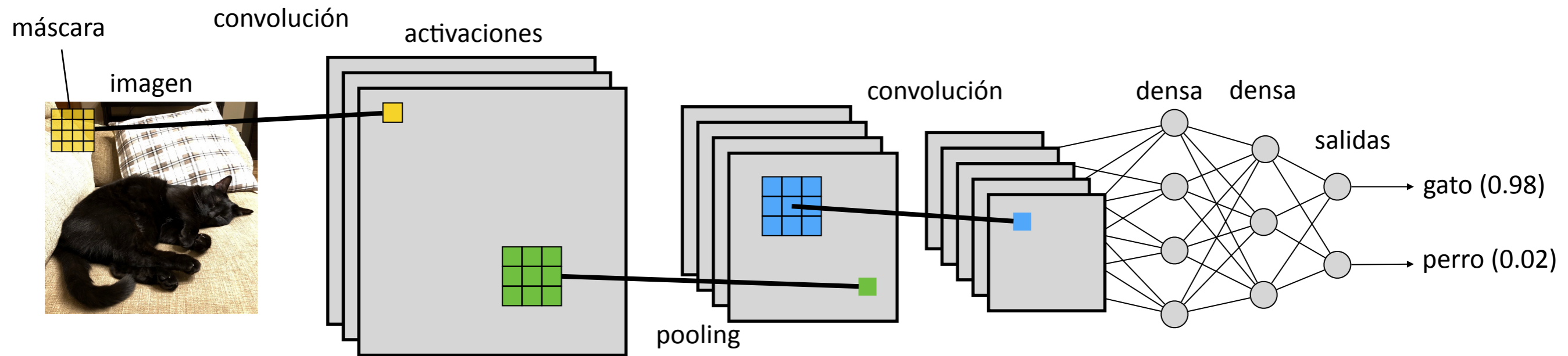
los pesos de la máscara se "aprenden"



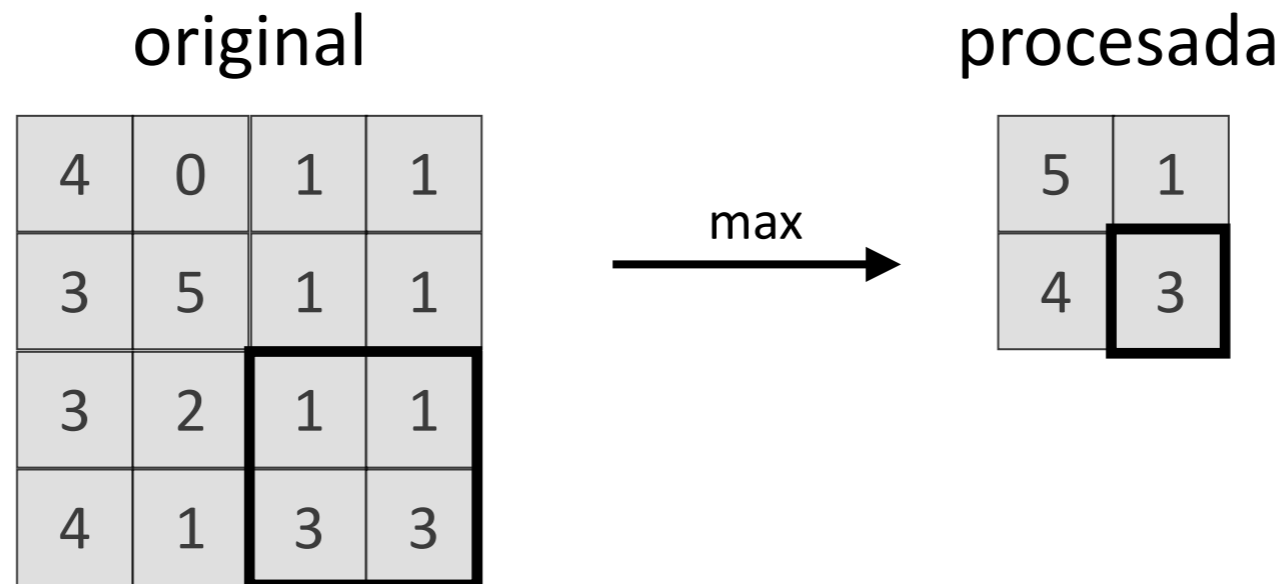
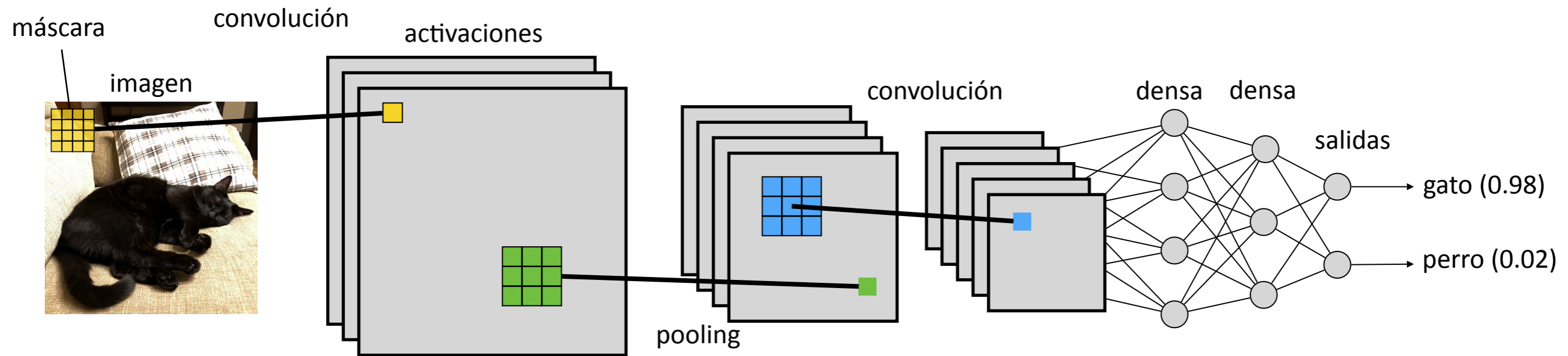
Operación de pooling



Operación de pooling



Operación de pooling



Operación de pooling

La operación de “pooling”
permite conseguir invarianza a traslaciones

Reduce el tamaño y los pesos
mejora generalización

Redes recurrentes

Concepto y red recurrente estándar

Cuando leemos,
la interpretación de cada palabra
se basa en las anteriores:

el corredor de Gijón llegó el **cuarto** a la meta
ayer estuve ordenando mi **cuarto**
no tengo un **cuarto**
partió el queso y guardó un **cuarto** para mí

Muchos problemas tienen "persistencia"
requieren información pasada
para interpretar la presente



procesos
con memoria

Redes recurrentes

Concepto y red recurrente estándar

Secuencia:

sucesión de valores

un valor puede depender de los anteriores

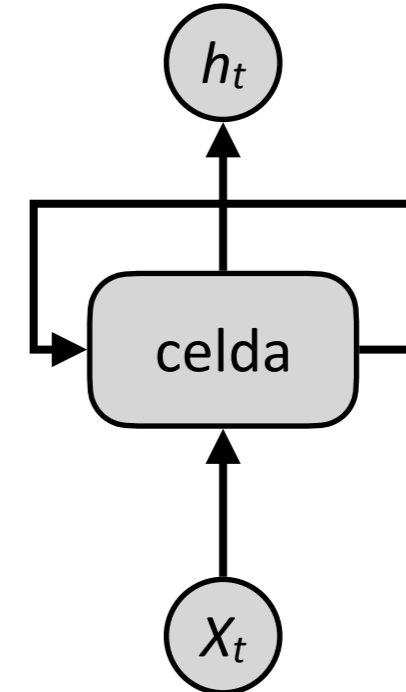
$$\{ \dots \mathbf{x}_{-1}, \mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n \dots \}$$

Redes recurrentes

tienen “bucles”,
reprocesan datos
que ellas mismas produjeron



consiguen que
la información persista

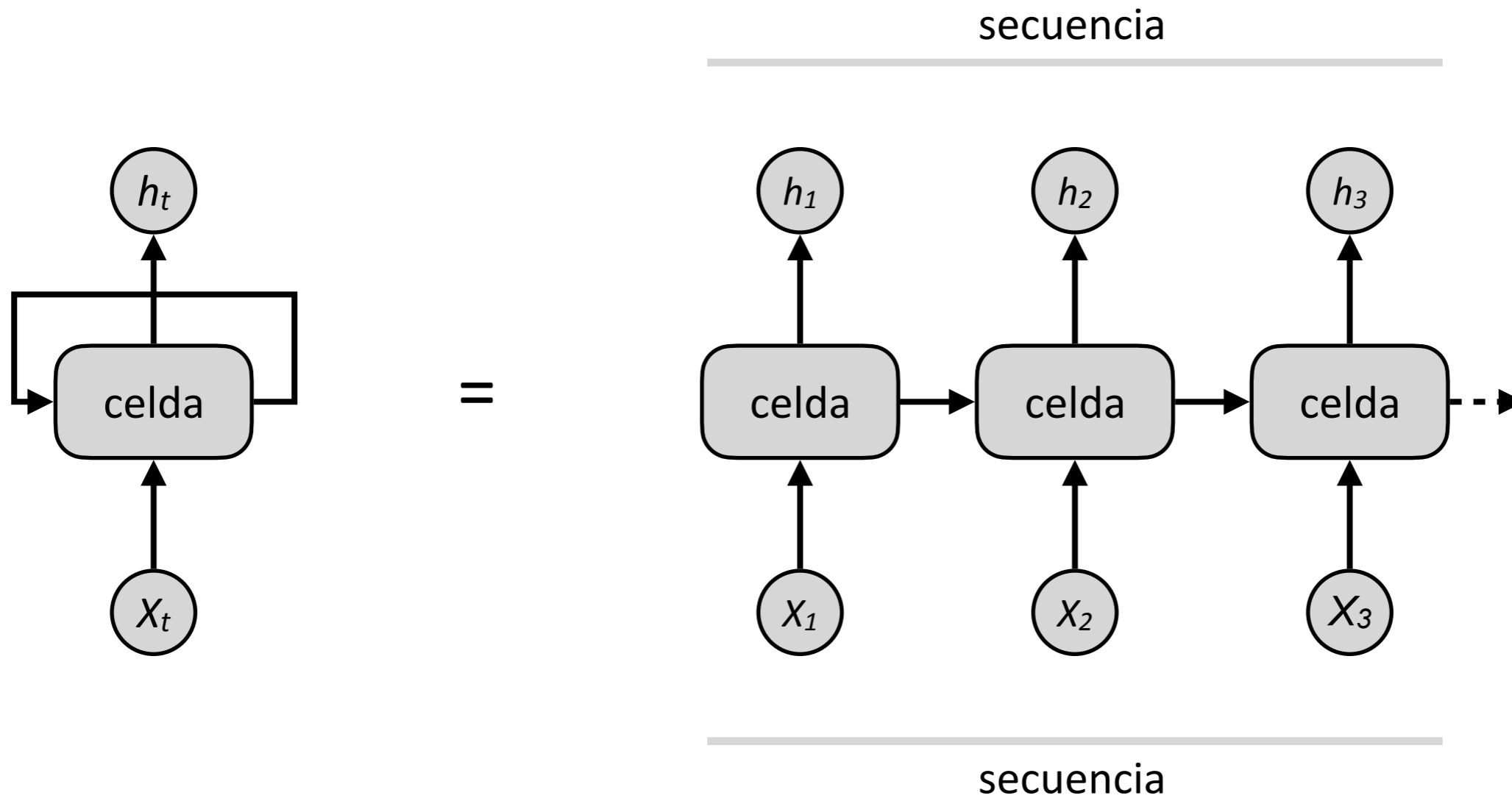


Referencia

Excelente artículo en el **blog de Colah**:
<http://colah.github.io/posts/2015-08-Understanding-LSTMs/>

Redes recurrentes

Concepto y red recurrente estándar



Referencia

Excelente artículo en el **blog de Colah**:
<http://colah.github.io/posts/2015-08-Understanding-LSTMs/>

Redes recurrentes

Concepto y red recurrente estándar

Ecuaciones dinámicas

$$s_{t+1} = f_{W,U}(s_t, x_t)$$

$$o_t = g_V(s_t)$$

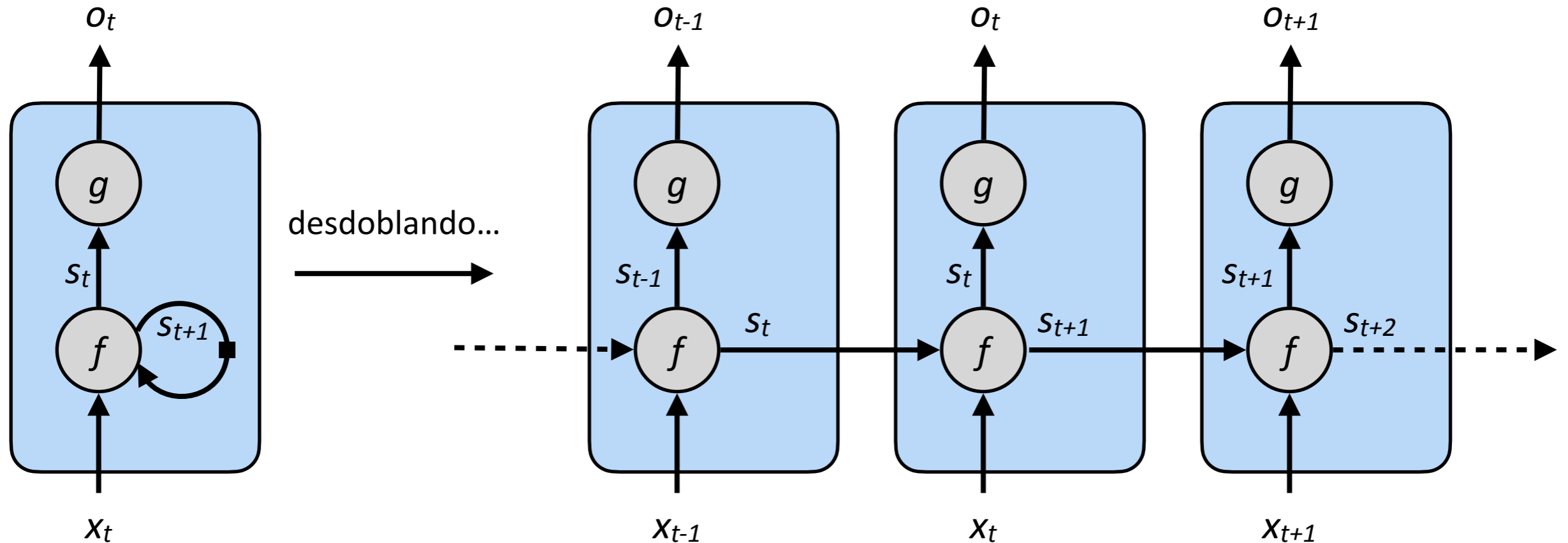


diagrama de bloques

Leyenda:

s_t : estado interno de la unidad

x_t : entrada

o_t : salida

W, U : parámetros de la función de estado $f()$

V : parámetros de la función de salida $g()$

representación equivalente

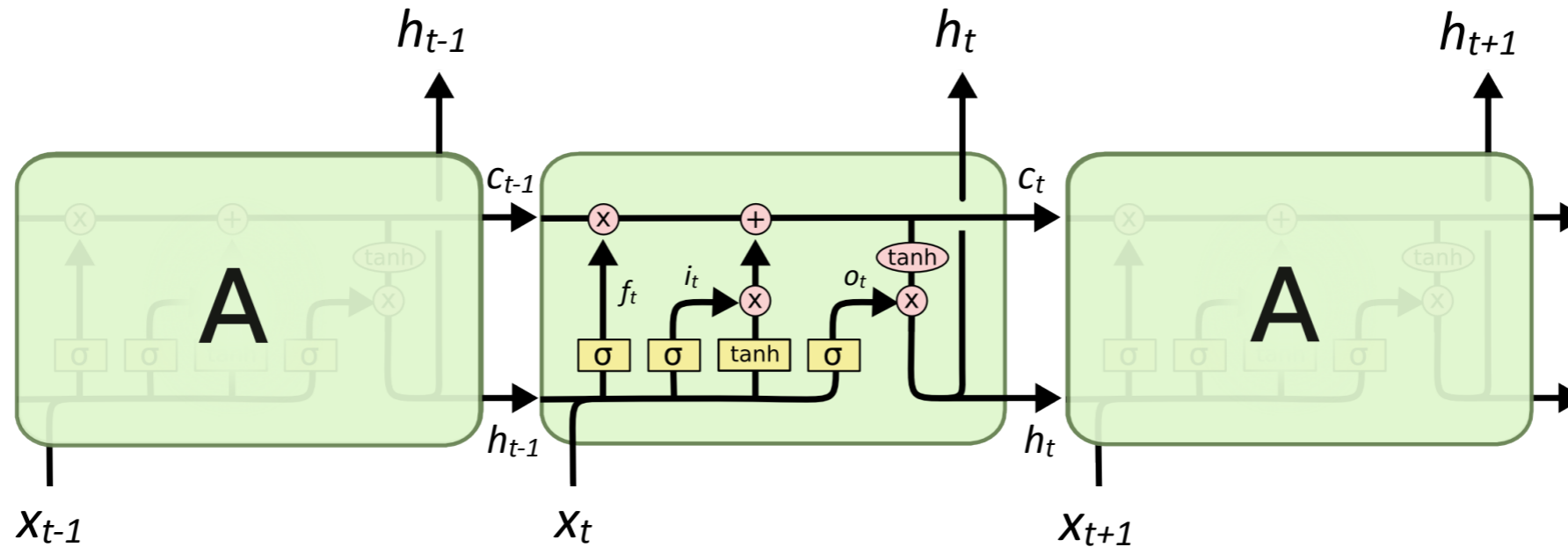
una unidad equivale a una red de profundidad infinita

tiene memoria

la salida depende del pasado

los pesos W, U definen una dinámica secuencial

W, U, V son entrenables



Fuente: figura adaptada del blog de Colah, <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>

Dinámica y estados internos

$$c_t = f_t c_{t-1} + i_t \tanh(W_c[h_{t-1}, x_t] + b_c)$$

$$h_t = o_t \tanh(c_t)$$

La información (estado) se transfiere hacia la derecha

Puertas (actúan como moduladores)

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad \text{entrada}$$

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \quad \text{olvido}$$

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \quad \text{salida}$$

Mediante “puertas” (“gates”) multiplicativas regula la transferencia de información

Referencia

Excelente artículo en el **blog de Colah**:

<http://colah.github.io/posts/2015-08-Understanding-LSTMs/>

Una LSTM tiene tres puertas

- entrada (relevancia de la entrada vs. estado)
- olvido (dinámica interna del estado)
- salida (qué parte dejamos fluir hacia la siguiente)

<http://isa.uniovi.es/GSDPI>

Elementos de una LSTM

Celda

- El estado de la celda se almacena en el vector c_t
- Es como una cinta transportadora
- Transporta información útil entre etapas
- la LSTM añade o quita información mediante "puertas"

Puertas

- Estructuras para dejar fluir o no la información
- Funcionan como grifos: cierran o abren paso a valores
- Una LSTM tiene tres puertas: forget, input, output

Puerta "forget"

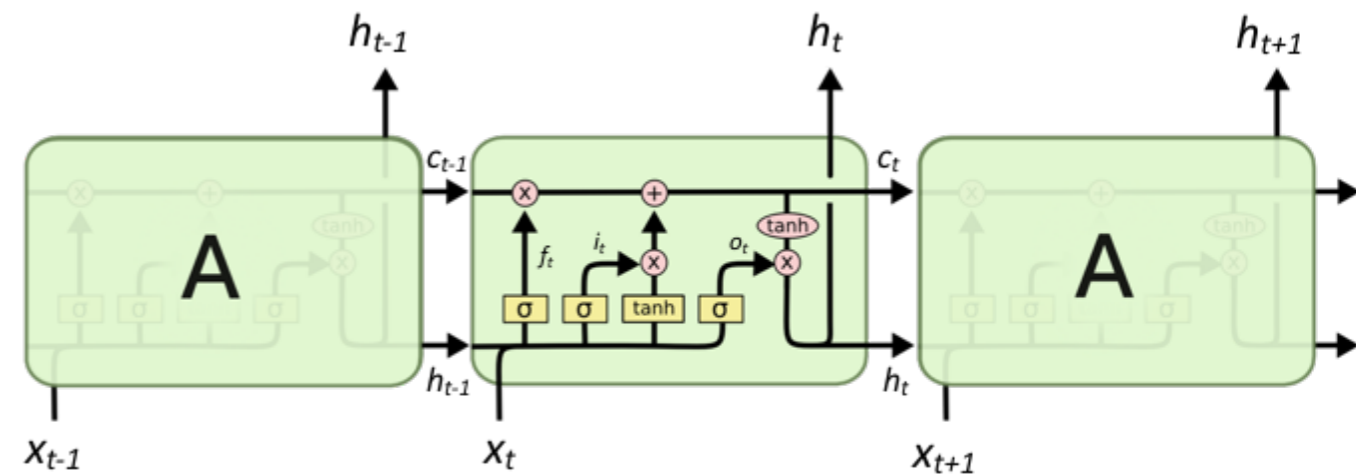
- decide cuánta información se conserva en la celda
- un "1" implica "conservar esto al 100%"
- un "0" implica "olvidarlo completamente"

Puerta "input"

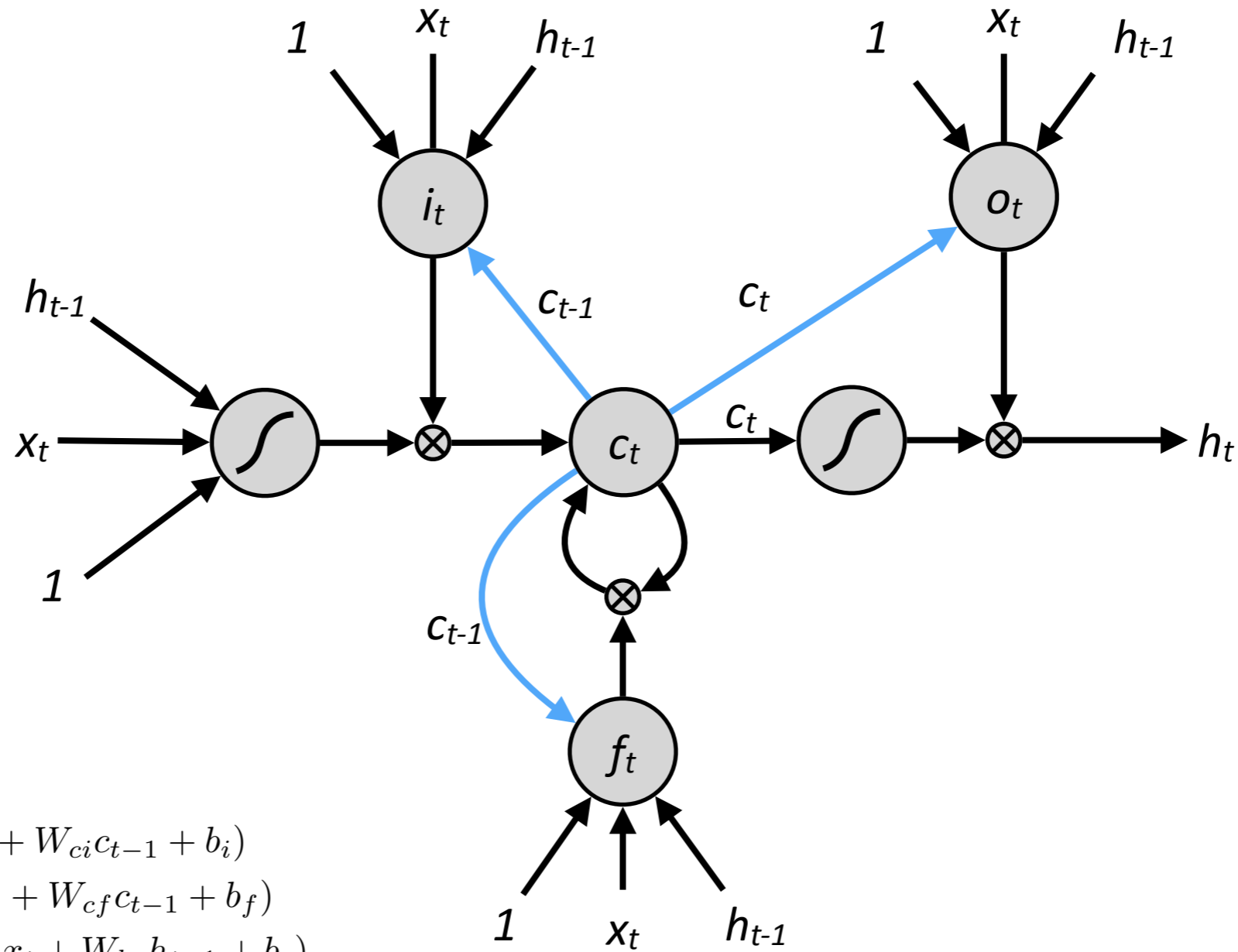
- decide qué información nueva (de x_t) se añade a la celda
- la tanh genera valores candidatos para cambiarla
- la sigmoide (grifo) decide qué valores candidatos lo hacen

Puerta "output"

- la tanh transforma valores de c_t en -1 a 1
- la sigmoide (grifo) decide qué información de c_t sale a h_t



Fuente: figura adaptada del blog de Colah, <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>



$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f)$$

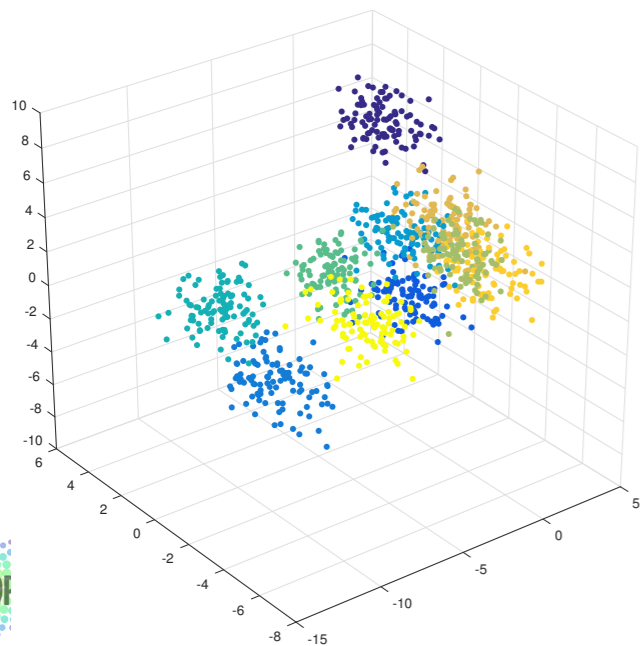
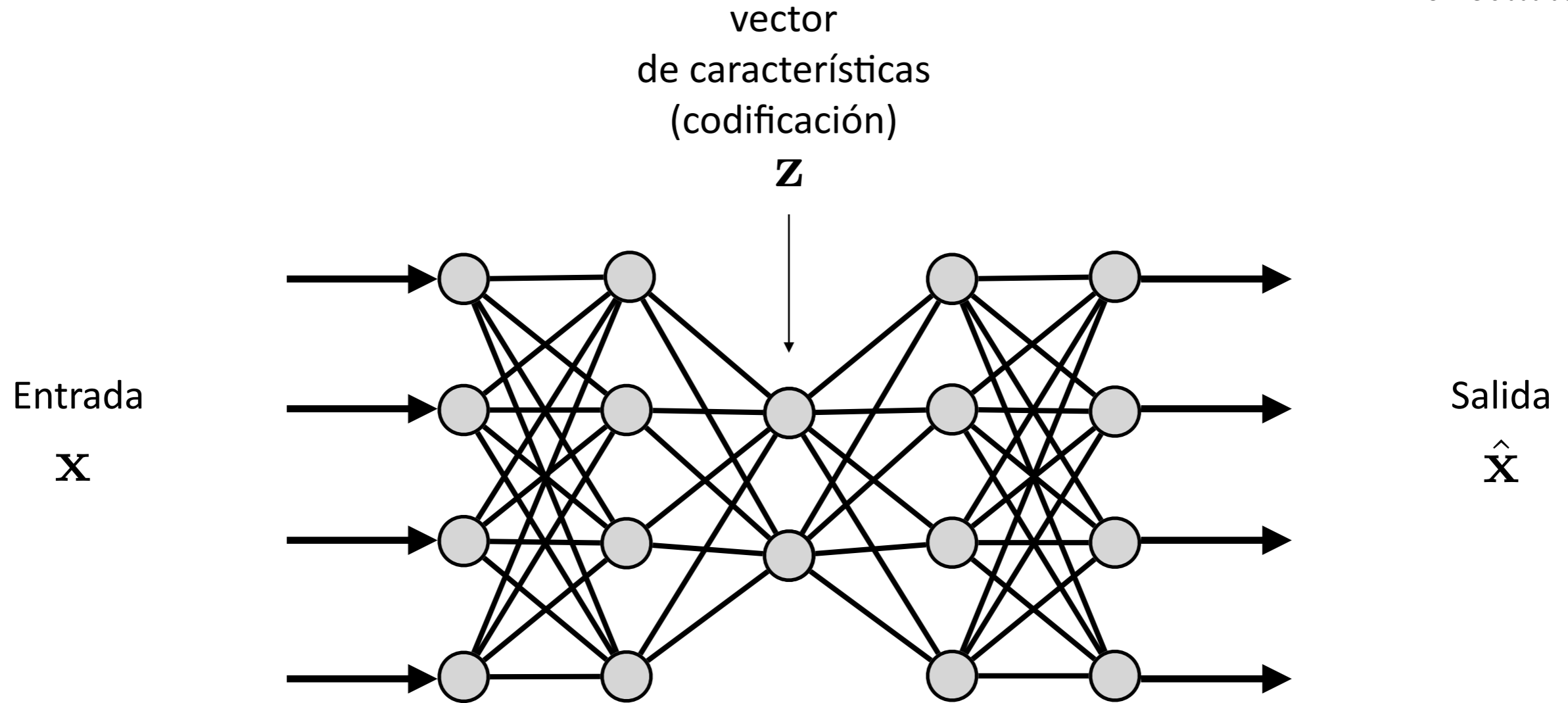
$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o)$$

$$h_t = o_t \tanh(c_t)$$

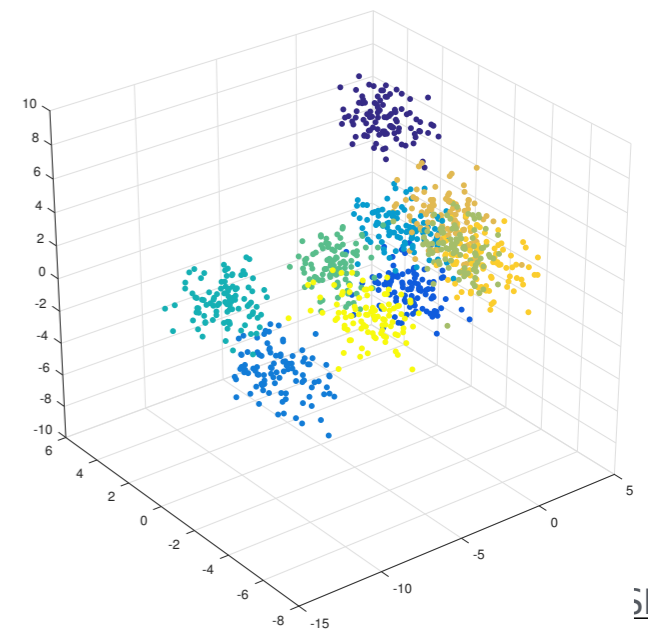
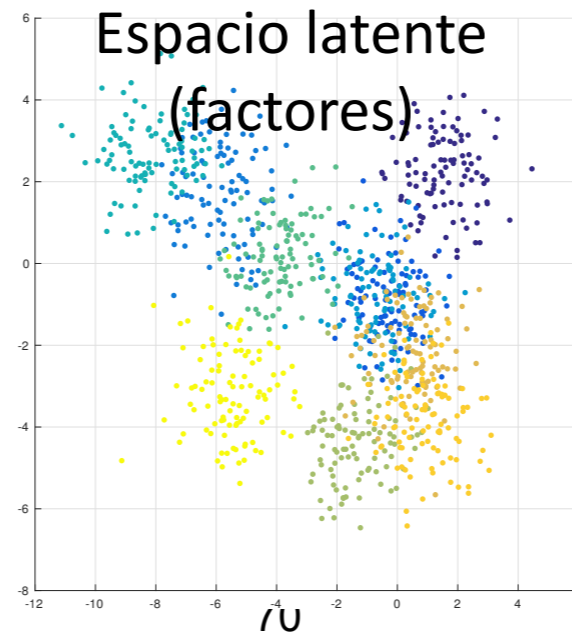
Referencia: A. Graves, N. Jaitly, A. Mohamed. **Hybrid Speech Recognition with Deep Bidirectional LSTM**

Deep autoencoders

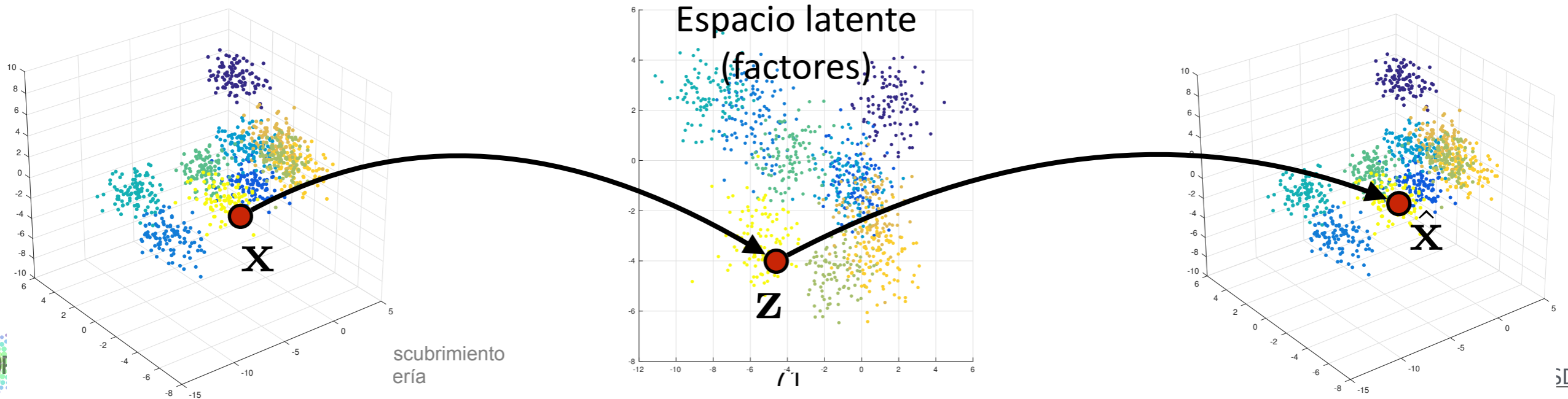
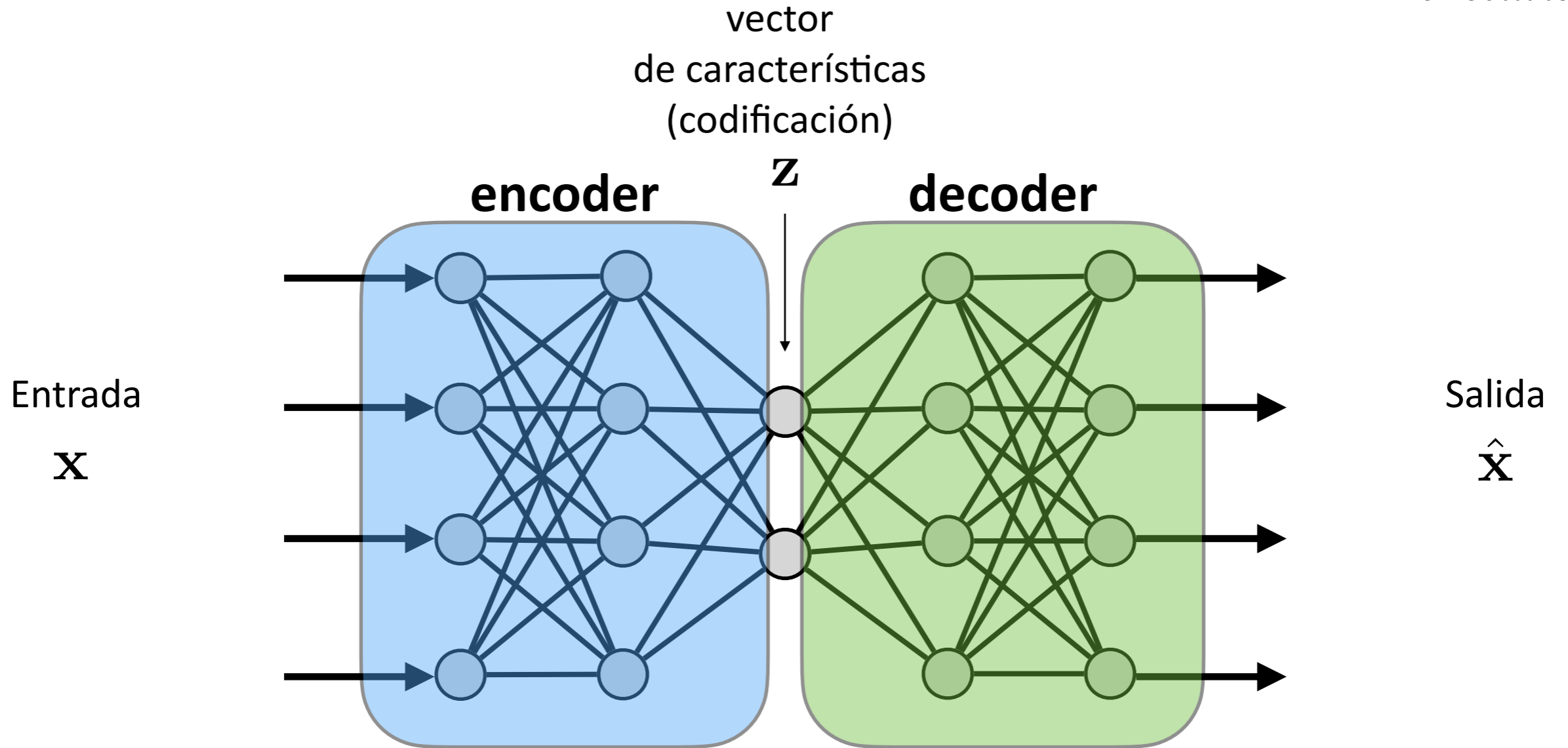


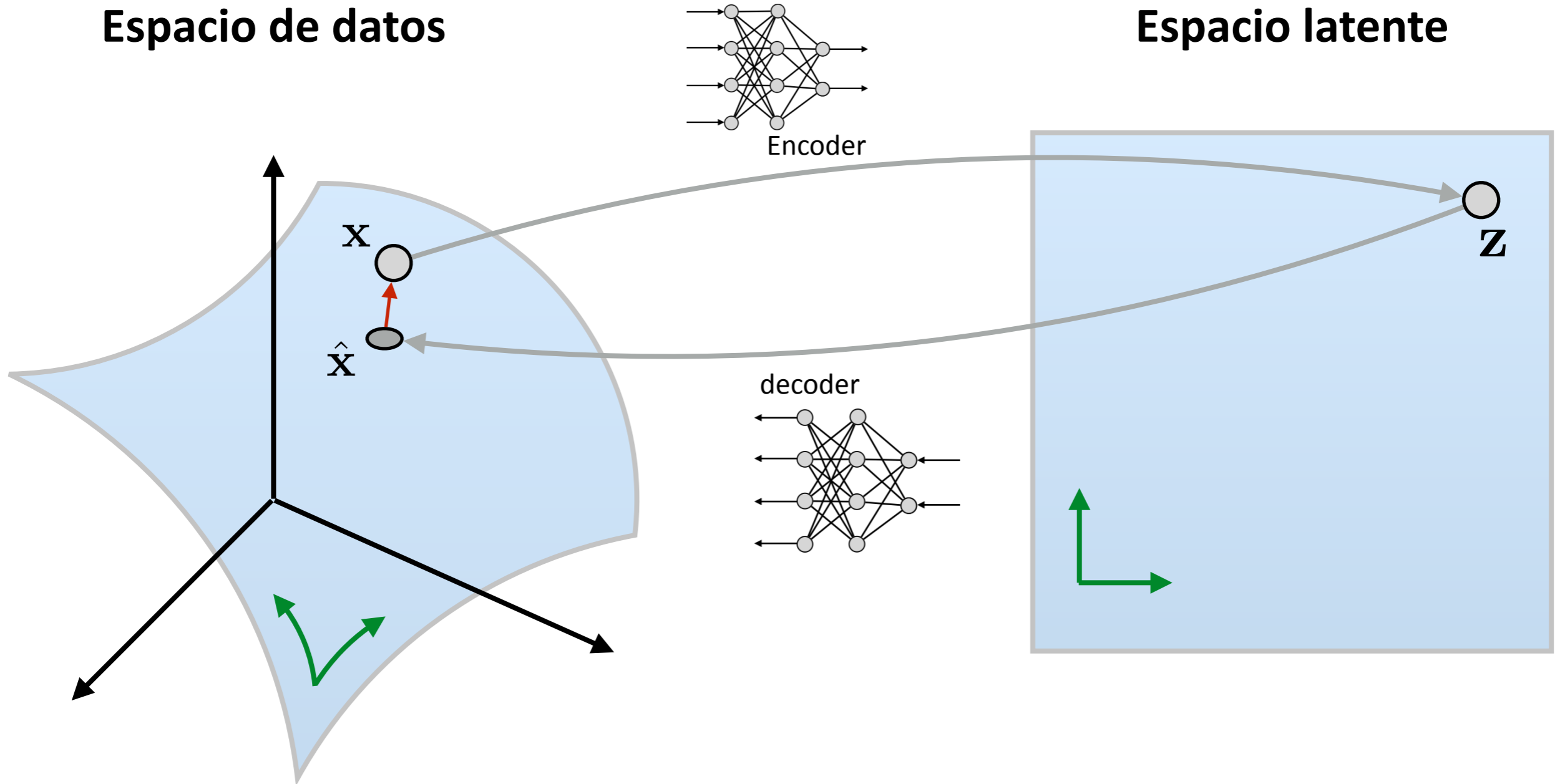
scubrimiento
ería

Espacio latente
(factores)



Deep autoencoders

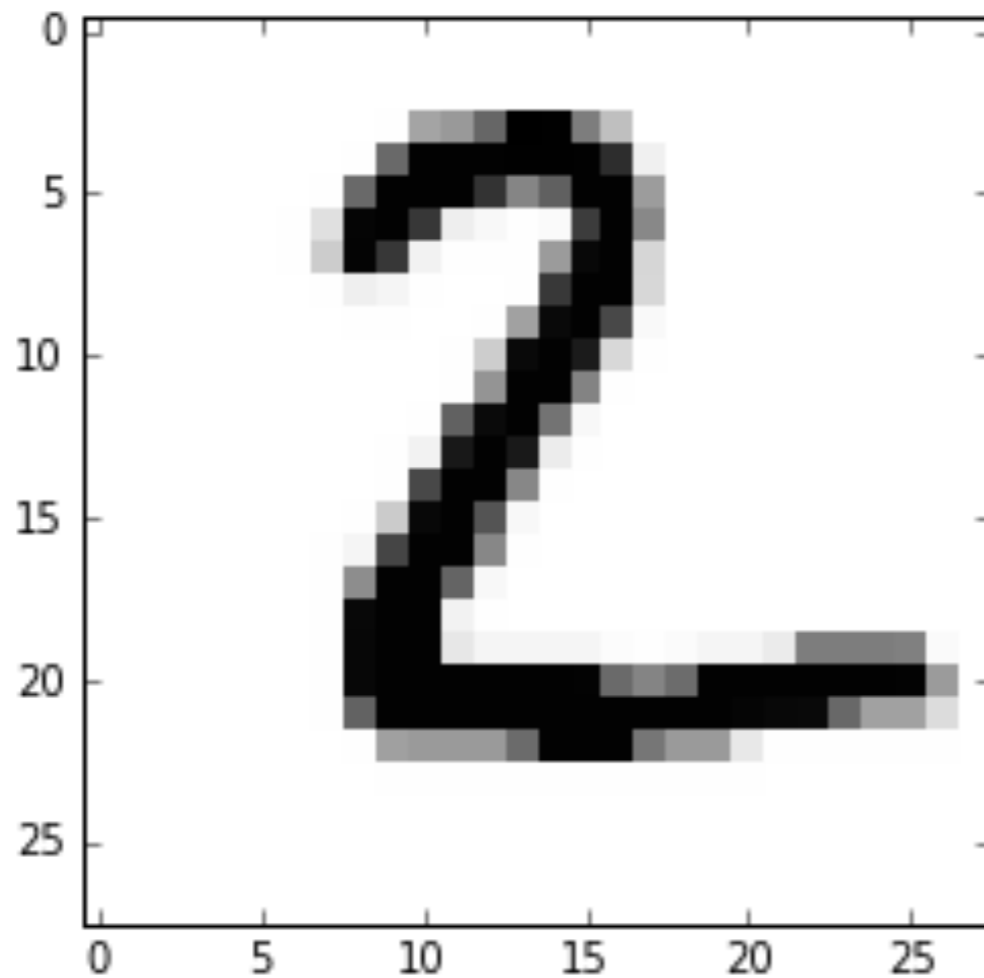




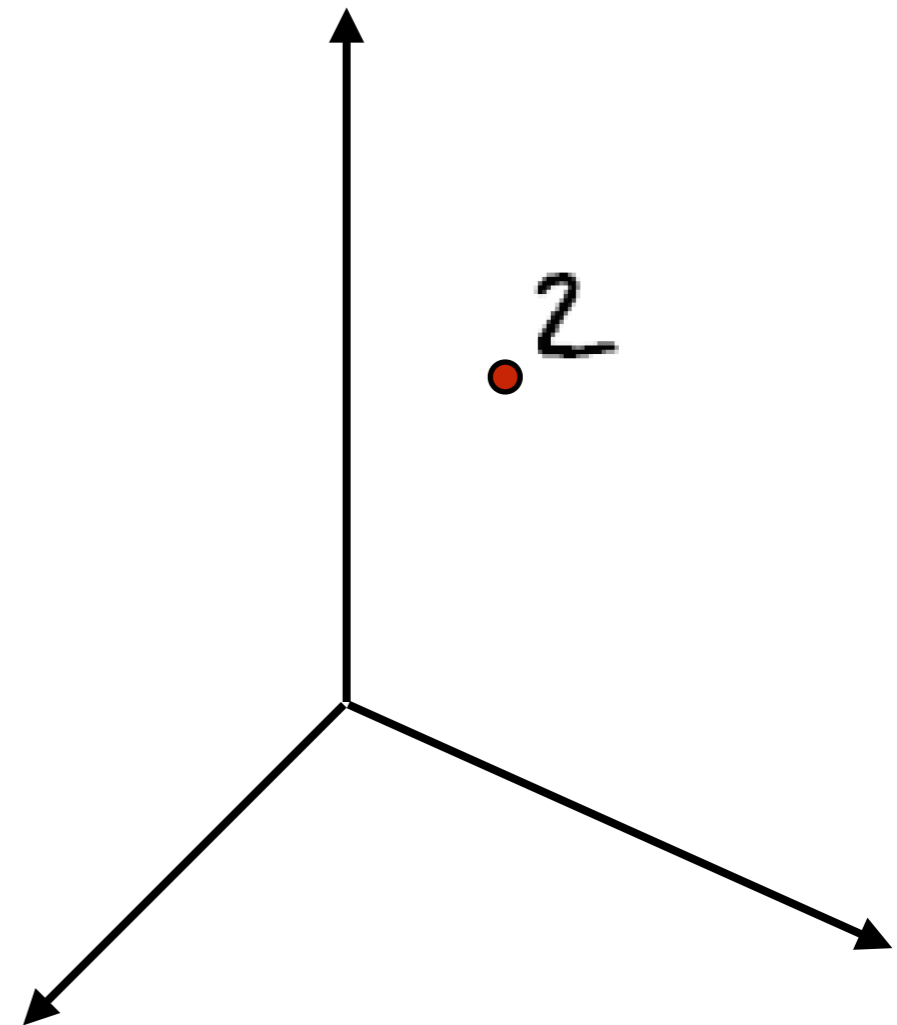
Ejemplo autoencoders

proyección de imágenes MNIST: dígitos manuscritos

imágenes de 28x28 px



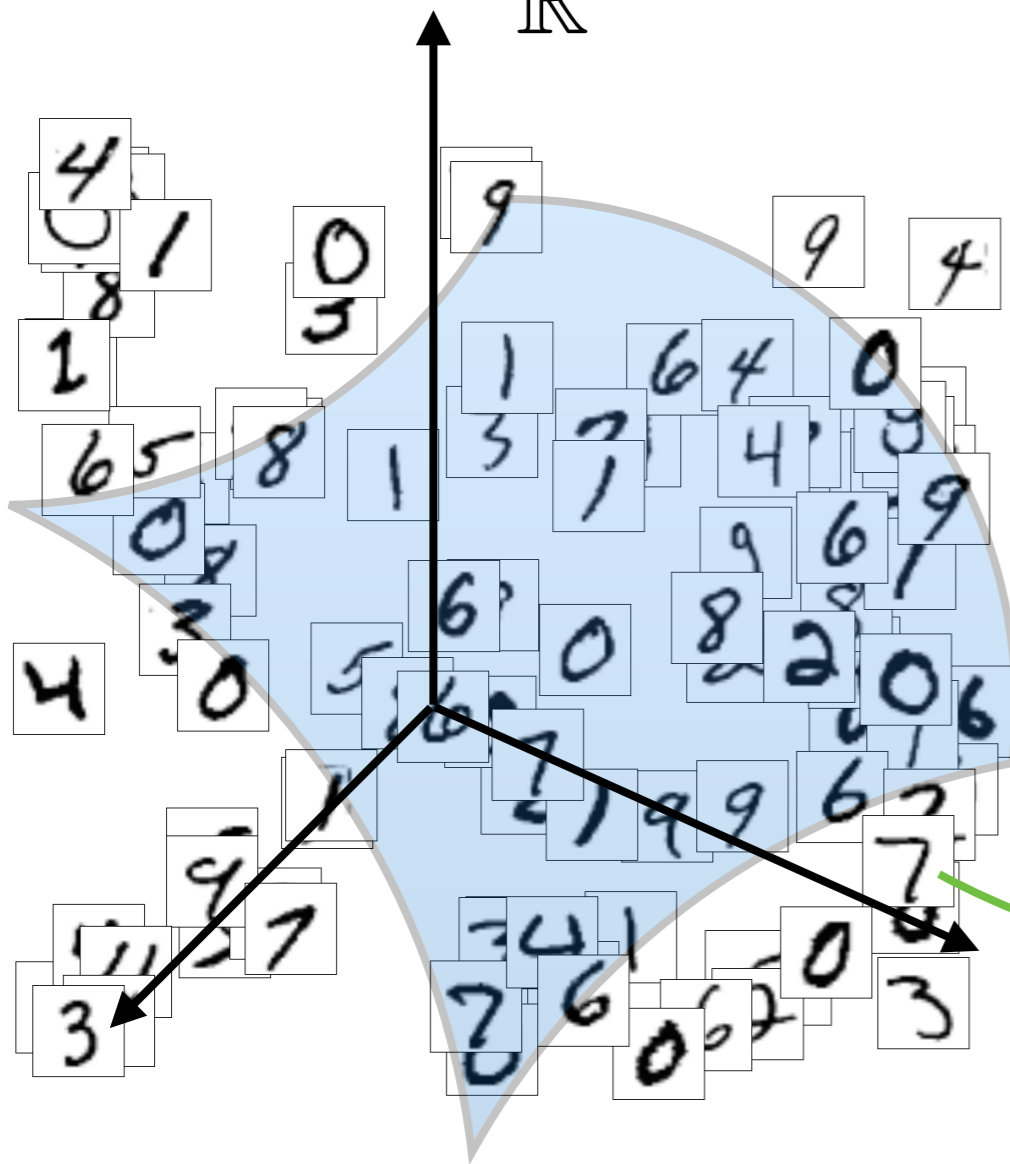
28x28 = 784 dimensiones



Espacio de datos

(alta dimensionalidad)

\mathbb{R}^{784}



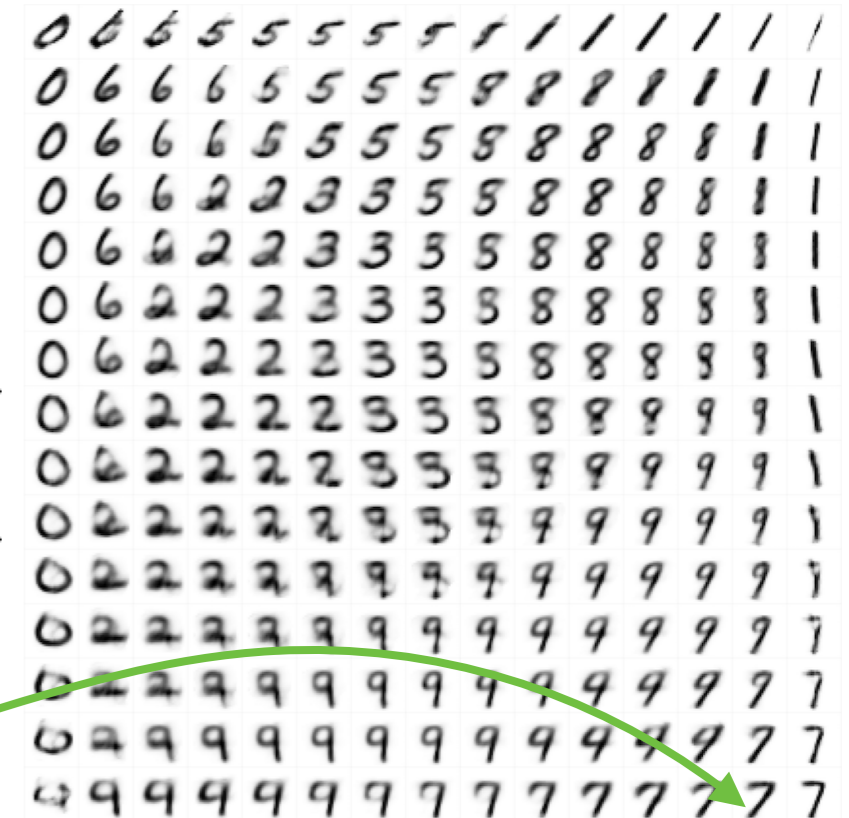
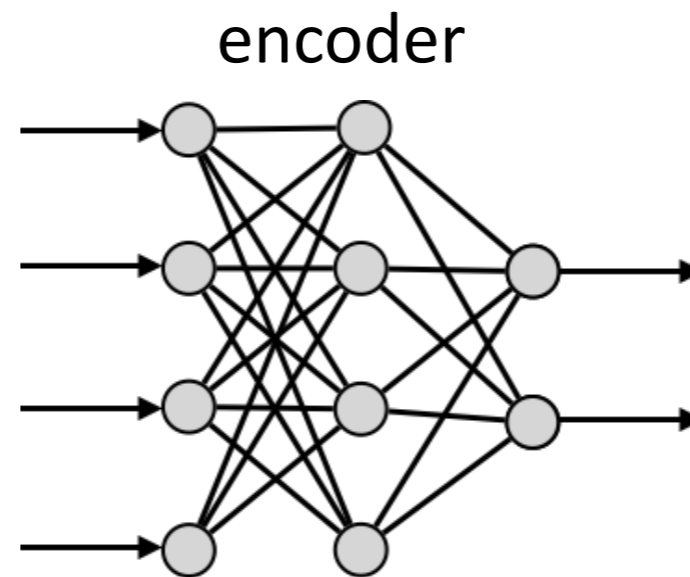
dataset MNIST:

60000 dígitos manuscritos

Espacio latente

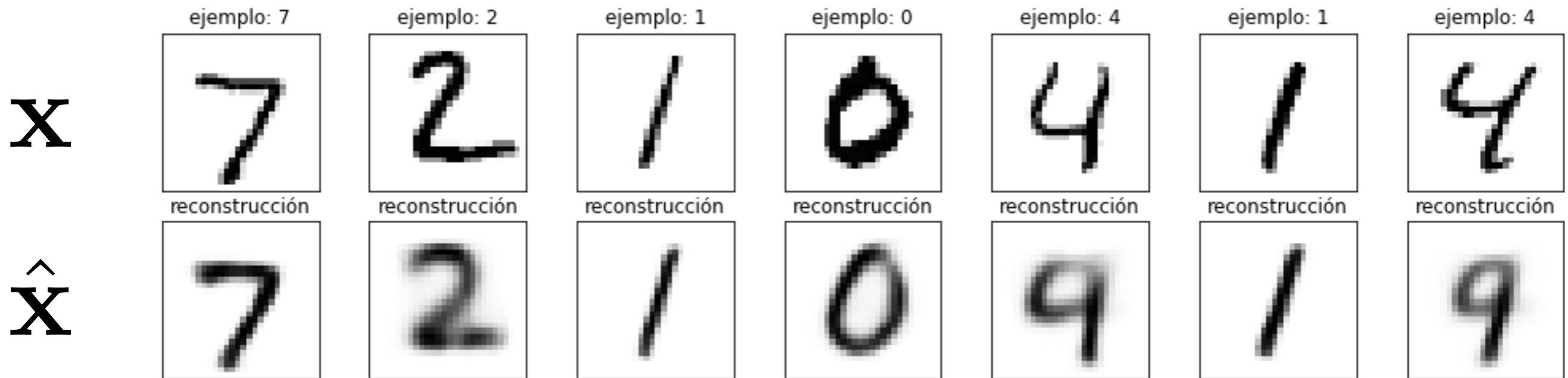
(baja dimensionalidad)

\mathbb{R}^2



Hemos creado
un “mapa visual”
de los datos

Reconstrucción de datos

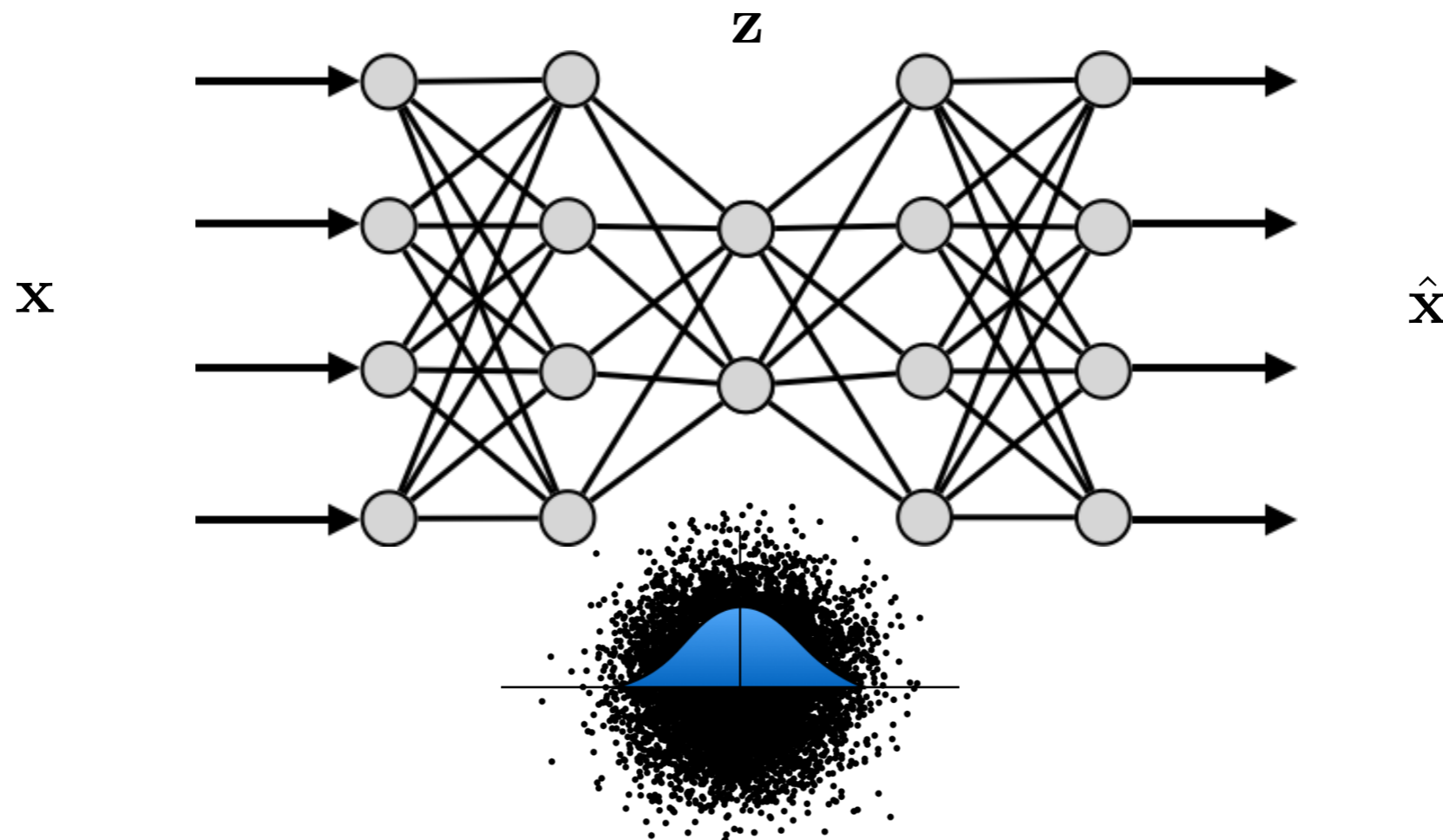


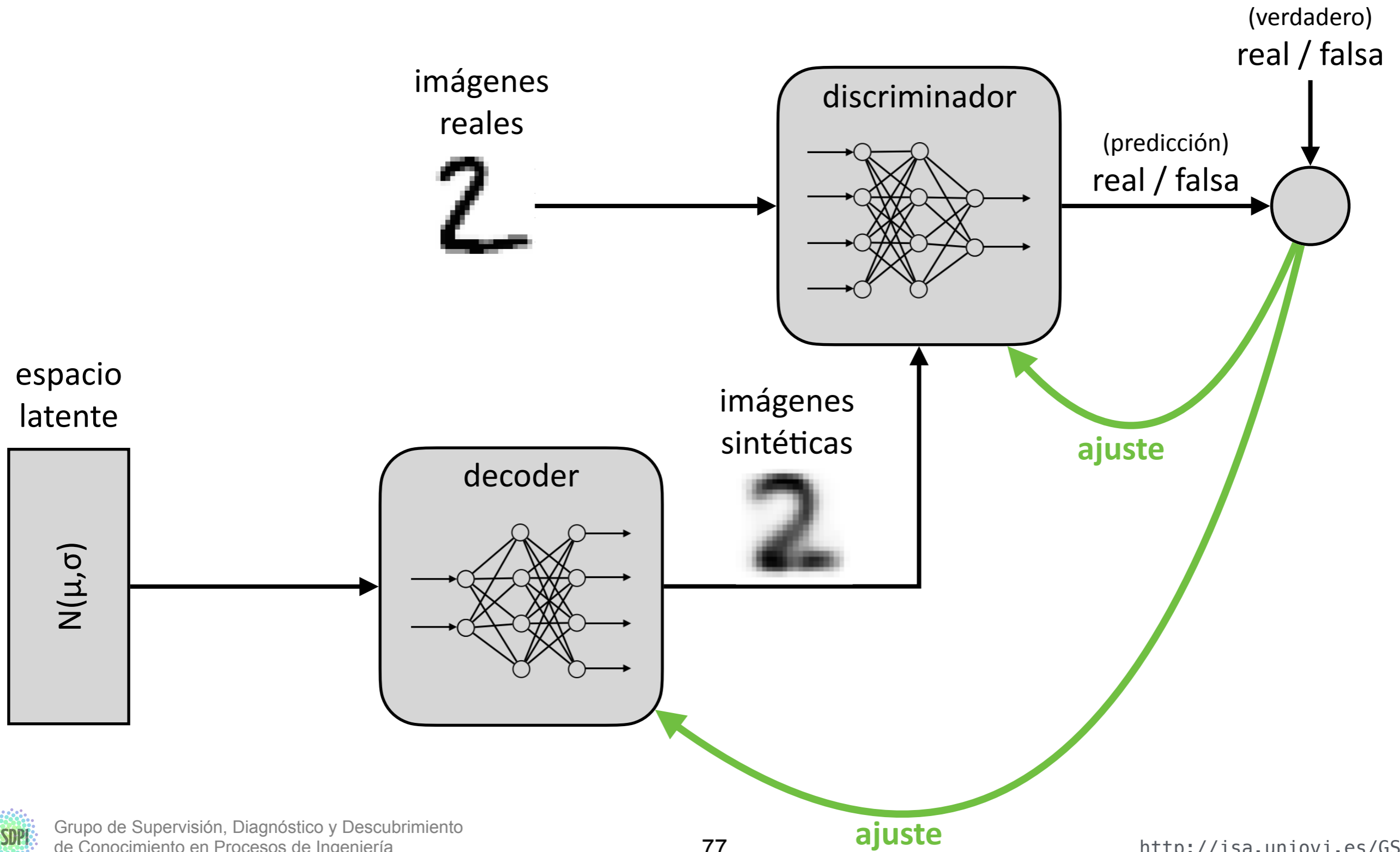
- reconstruir ejemplos corruptos con ruido
- generar ejemplos “creíbles”
- detección de fallos / detección de fraude \rightarrow residuos $r = \|\mathbf{x} - \hat{\mathbf{x}}\|^2$

1) Impone restricción en espacio latente
forzamos gaussiana $N(0,1)$

→ si genero un dato $N(0,1)$ → produce imagen “creíble”

2) inyecta ruido durante el aprendizaje → mejor generalización





Reconstrucción

Reconstruir imágenes o audio corruptos

Detección de novedades

detección de fallos

detección de fraude

Visualización de datos

generación automática de “mapas” 2D y 3D

de datos complejos (imágenes, audio, vibraciones, etc.)

Modelos generativos

generación de ejemplos sintéticos

(imágenes, música, etc.) <https://goo.gl/s49MPU>

los ejemplos son “creíbles”: siguen las reglas de los ejemplos reales

“How to Generate Music using a LSTM Neural Network in Keras”

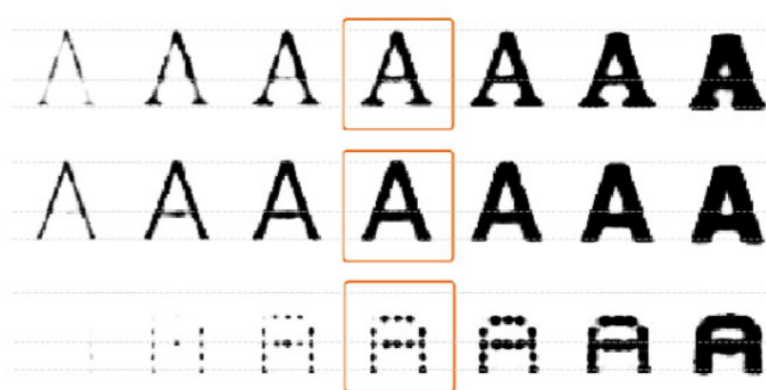
<https://distill.pub/2017/aia/>

Dec. 4, 2017

COMMENTARY

Using Artificial Intelligence to Augment Human Intelligence

By creating user interfaces which let us work with the representations inside machine learning models, we can give people new tools for reasoning.



Generación de ejemplos

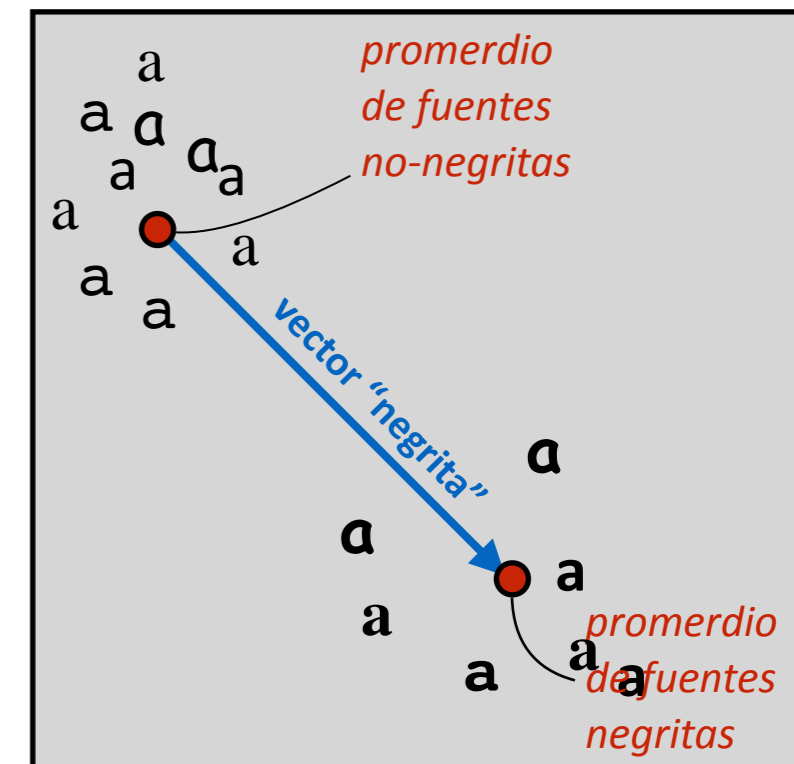
fuentes, caras, frases, moléculas ...

Idea de “vectores de atributos”

direcciones en el espacio latente

muestran atributos como “negrita”

u otras más abstractas como “sonrisa”, “femenino”, etc.



Transfer Learning o aprendizaje inductivo

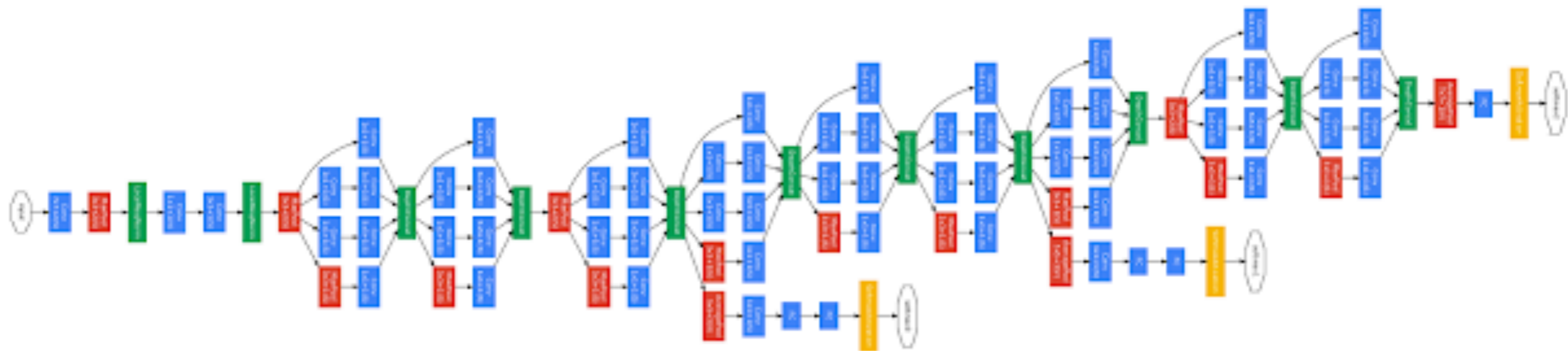
“almacenar conocimiento obtenido resolviendo un problema y aplicarlo a otro problema diferente pero relacionado”

Idea:

1. Entrenar una red “monstruosa”,
muy **profunda**
con millones de pesos,
sobre millones de imágenes ...
2. Aplicarla con cambios menores
a un problema relacionado:
ej. → clasificar radiografías

Existen modelos preentrenados

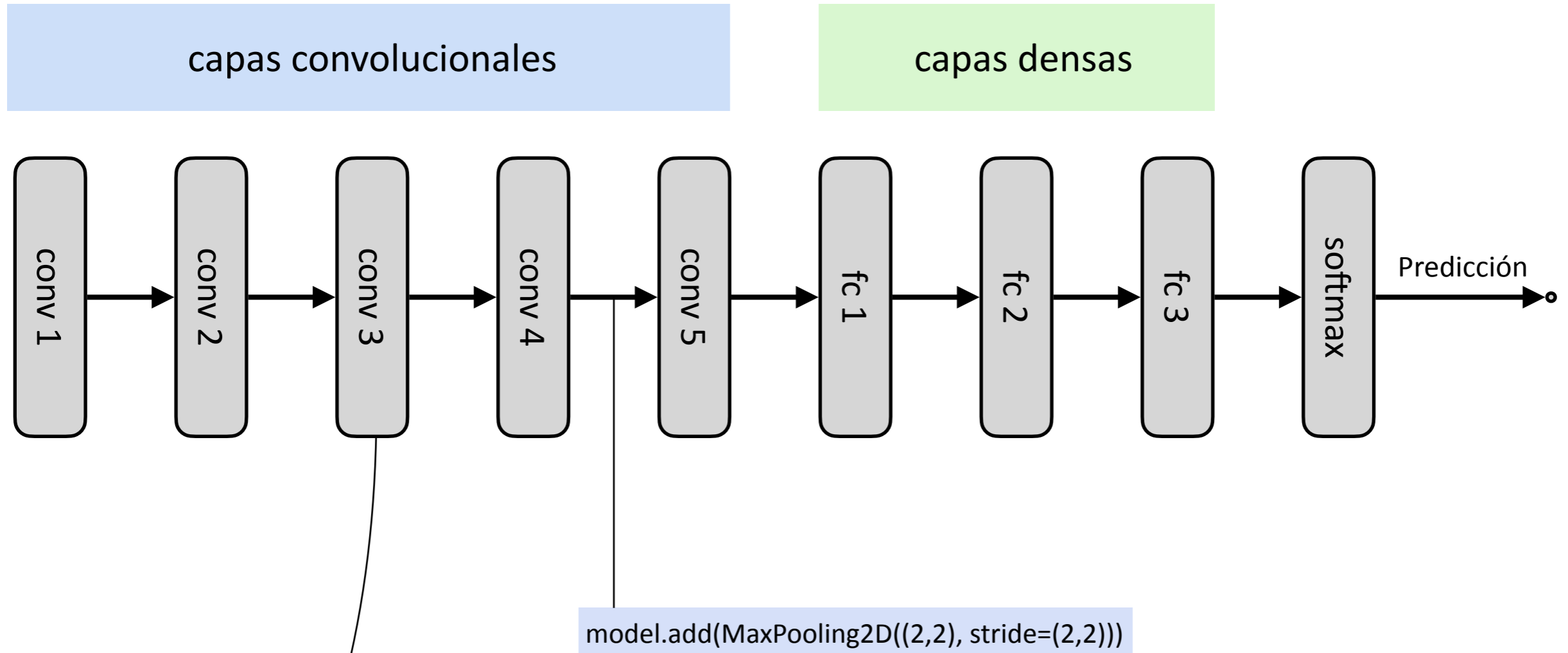
- Oxford VGG Model
- GoogLeNet
- Google Inception Model
- Microsoft ResNet Model
- Google's word2vec Model
- Stanford's GloVe Model
- Caffe Model Zoo



Arquitectura GoogLeNet

fuelle: "Using Machine Learning to Explore Neural Network Architecture"

<https://goo.gl/c6vMb4>



```
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(256, 3, 3, activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(256, 3, 3, activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(256, 3, 3, activation='relu'))
```

Artículo original

Very Deep Convolutional Networks for Large-Scale Image Recognition
K. Simonyan, A. Zisserman
arXiv:1409.1556

Algunos ejemplos y aplicaciones de deep learning

RESEARCH

Generating Focused Molecule Libraries for Drug Discovery with Recurrent Neural Networks

Marwin HS Segler^{1*}, Thierry Kogej², Christian Tyrchan³ and Mark P Waller⁴

Abstract

In *de novo* drug design, computational strategies are used to generate novel molecules with good affinity to the desired biological target. In this work, we show that recurrent neural networks can be trained as generative models for molecular structures, similar to statistical language models in natural language processing. We demonstrate that the properties of the generated molecules correlate very well with the properties of the molecules used to train the model. In order to enrich libraries with molecules active towards a given biological target, we propose to fine-tune the model with small sets of molecules, which are known to be active against that target.

Against *Staphylococcus aureus*, the model reproduced 14% of 6051 hold-out test molecules that medicinal chemists designed, whereas against *Plasmodium falciparum* (Malaria) it reproduced 28% of 1240 test molecules. When coupled with a scoring function, our model can perform the complete *de novo* drug design cycle to generate large sets of novel molecules for drug discovery.

Keywords: computer-assisted drug design; recurrent neural networks

1 Introduction

Chemistry is the language of nature. Chemists speak it fluently and have made their discipline one of the true contributors to human well-being, which has “change[d] the way you live and die”.^[1] This is particularly true for medicinal chemistry. However, creating novel drugs is an extraordinarily hard and complex problem.^[2] One of the many challenges in drug design is the sheer size of the search space for novel molecules. It has been estimated that 10^{60} drug-like molecules could possibly be synthetically accessible.^[3] Chemists have to select and examine molecules from this large space to find molecules that are active towards a biological target. Active means for example that a molecule binds to a biomolecule, which causes an effect in the living organism, or inhibits replication of bacteria. Modern high-throughput screening techniques allow to test molecules in the order of 10^6 in the lab.^[4] However, larger experiments will get prohibitively expensive. Given this practical limitation of *in vitro* experiments, it is desirable to have computational tools to narrow down the enormous search space. *Virtual screening* is a commonly used strategy to search for promising molecules amongst mil-

lions of existing or billions of virtual molecules.^[5] Searching can be carried out using similarity-based metrics, which provides a quantifiable numerical indicator of closeness between molecules. In contrast, in *de-novo* drug design, one aims to directly create novel molecules that are active towards the desired biological target.^[6, 7] Here, like in any molecular design task, the computer has to

- i create molecules,
- ii score and filter them, and
- iii search for better molecules, building on the knowledge gained in the previous steps.

Task i, the generation of novel molecules, is usually solved with one of two different protocols.^[7] One strategy is to build molecules from predefined groups of atoms or fragments. Unfortunately, these approaches often lead to molecules that are very hard to synthesise.^[8] Therefore, another established approach is to conduct virtual chemical reactions based on expert coded rules, with the hope that these reactions could then also be applied in practice to make the molecules in the laboratory.^[9] These systems give reasonable drug-like molecules, and are considered as “the solution” to the structure generation problem.^[2] We generally share this view. However, we have recently shown that the predicted reactions from these rule-based expert systems can sometimes fail.^[10] Also, fo-

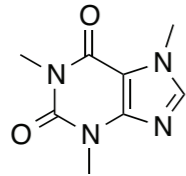
*Correspondence: marwin.segler@uni-muenster.de

¹Institute of Organic Chemistry & Center for Multiscale Theory and Computation, Westfälische Wilhelms-Universität, Münster, Germany
Full list of author information is available at the end of the article

1. Codifica moléculas como secuencias utilizando un procedimiento estándar (*SMILES*)
2. Entrena una RNN para predecir el siguiente símbolo en una secuencia larga
La red RNN “aprende” la gramática de moléculas válidas
3. Ajuste fino con conjunto reducido de medicamentos (analgésicos comunes)
4. A partir de una semilla la RNN genera recursivamente secuencias de medicamentos nuevas y con las propiedades de los analgésicos
5. De las secuencias se obtienen las moléculas nuevas. ¡Las propiedades físicoquímicas fueron similares!

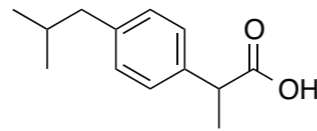
arXiv:1701.01329v1 [cs.NE] 5 Jan 2017

Codificación "SMILES" (molecula → secuencia)



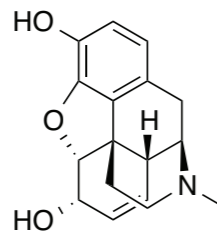
Caffeine

CN1c2ncn(C)c2C(=O)N(C)C1=O



Ibuprofen

CC(C)Cc1ccc(cc1)C(C)C(=O)O



Morphine

[H][C@]12C=C[C@H](O)[C@@H]3O[C@H]4[C@@H](O)CC[C@@]235)ccc4O

Conjunto de datos

ChEMBL

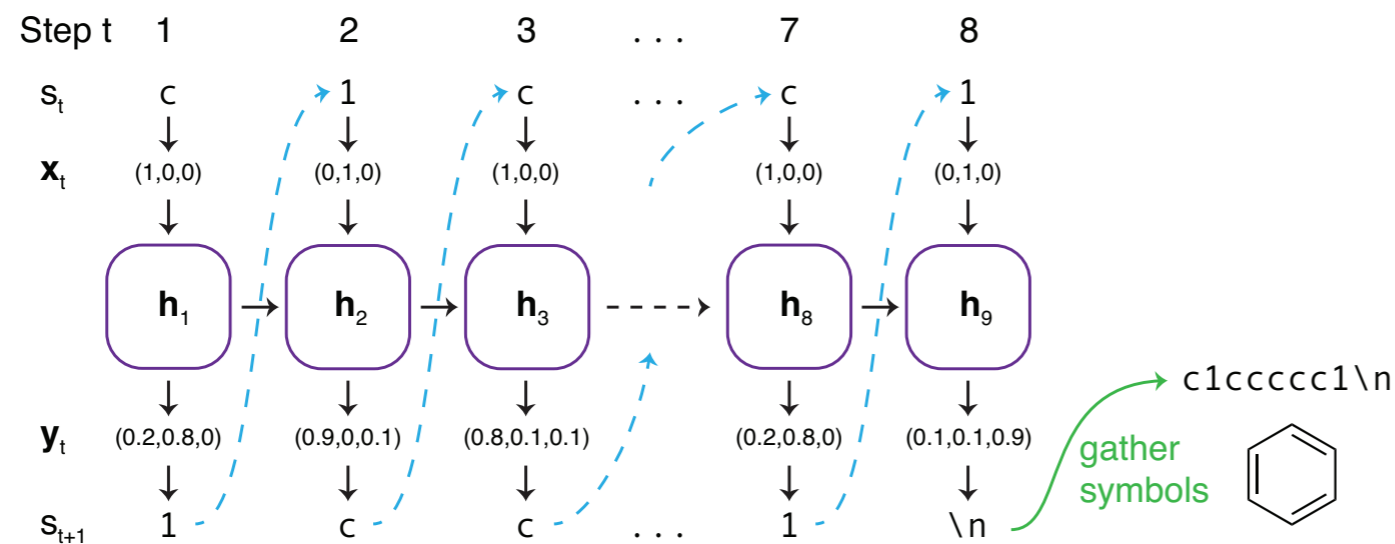
72M caracteres

vocabulario de 51 símbolos

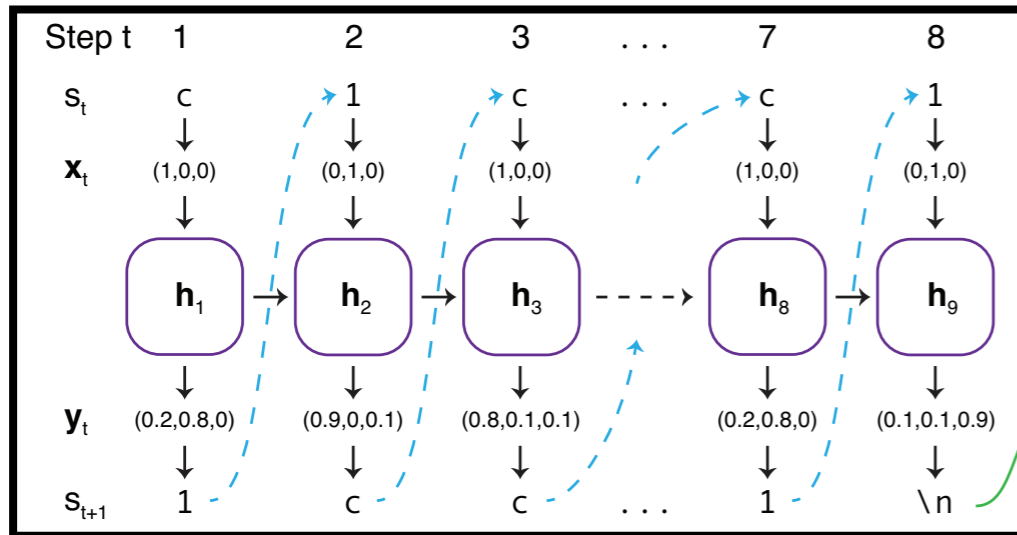
Transfer learning

1. Entrenar para otra tarea en la que hay muchos ejemplos
2. Entrenar el conjunto más pequeño (fine tuning)

RNN "predecir el siguiente símbolo" de una secuencia hasta el terminador "\n"



RNN genera secuencia recursivamente a base de predecir el siguiente símbolo

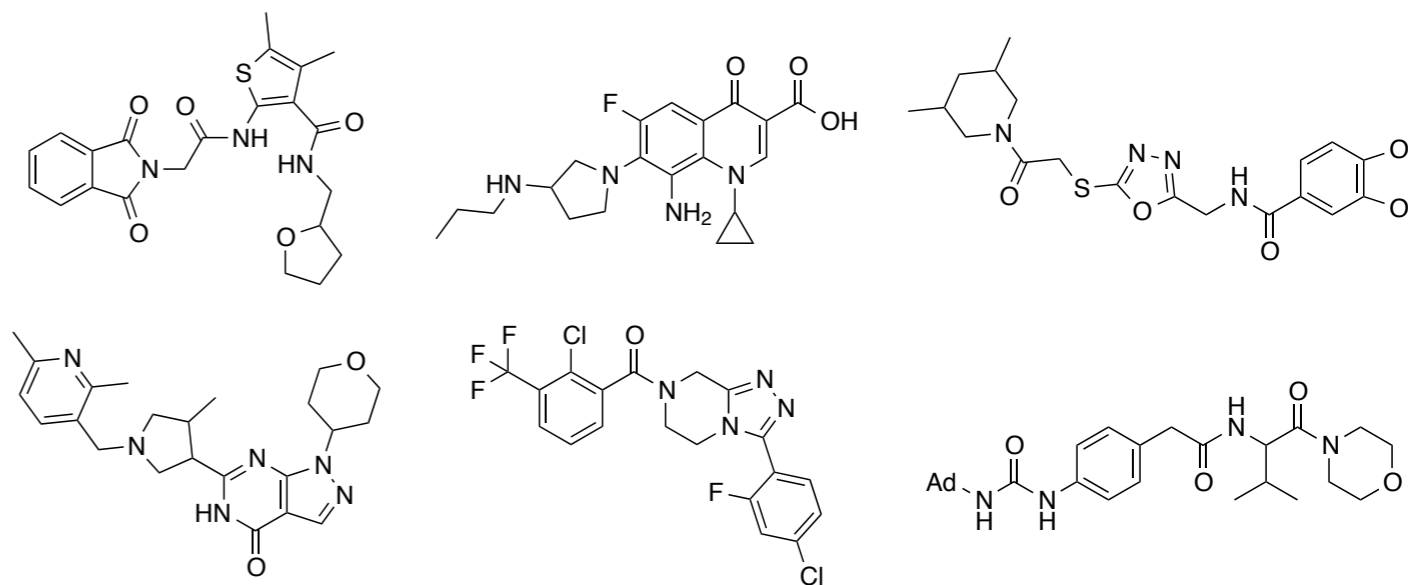


A partir del batch 3000 empieza a generar moléculas "válidas"

Batch	Generated Example	valid
0	<chem>Oc.BK5i%ur+7oAFc7L3T=F8B5e=n)CS6RCTAR((OVCp1CApb)</chem>	no
1000	<chem>OF=CCC20CCCC)C2)C1CNC2CCCCCCCCCCCCCCCCCCCCCCC</chem>	no
2000	<chem>O=C(N)C(=O)N(c1occc1OC)c2ccccc2OC</chem>	yes
3000	<chem>O=C1C=2N(c3cc(ccc3OC2CCC1)CCCc4cn(c5c(C1)cccc54)C)C</chem>	yes

Recupera moléculas a partir de las secuencias SMILE

Las moléculas resultantes fueron similares a las de entrenamiento en 7 descriptores físico-químicos



Aplicaciones de DL

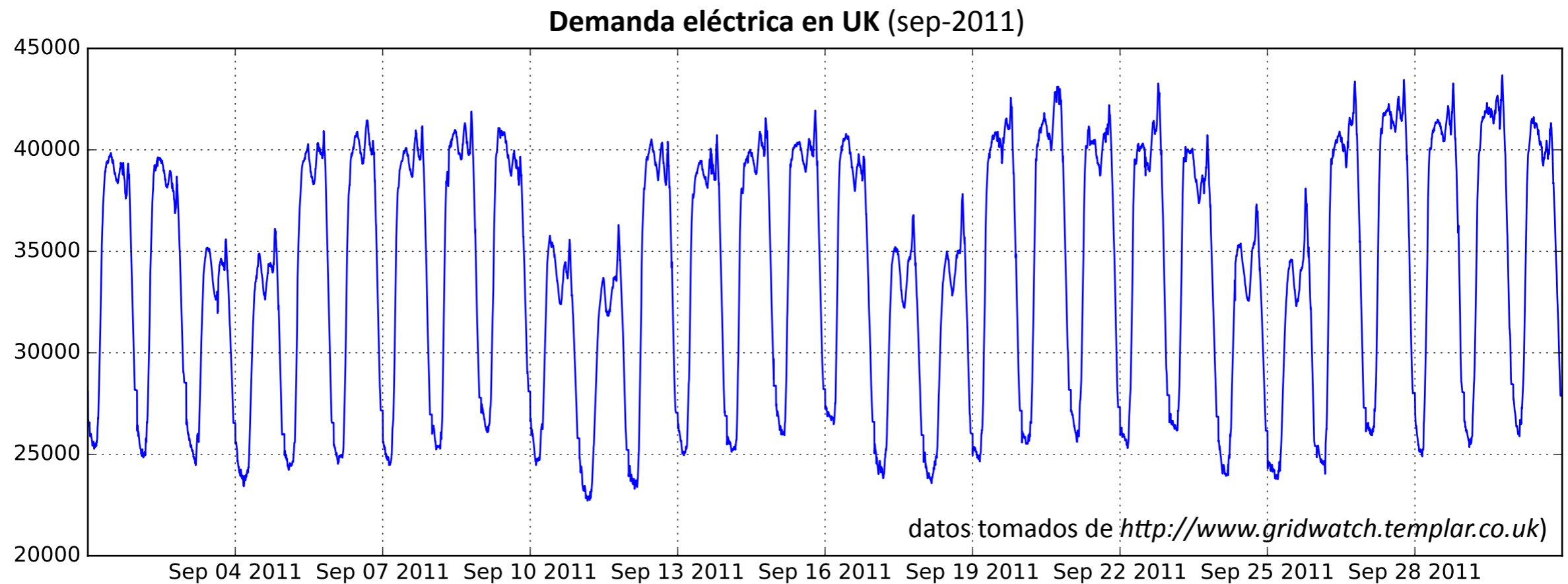
Predicción de demanda eléctrica

Objetivo

predecir la demanda eléctrica
en las 24 h siguientes

... a partir de

demanda de las últimas 48 horas (N=48)
semana del año (W)
día de la semana (D)
hora del día (H)

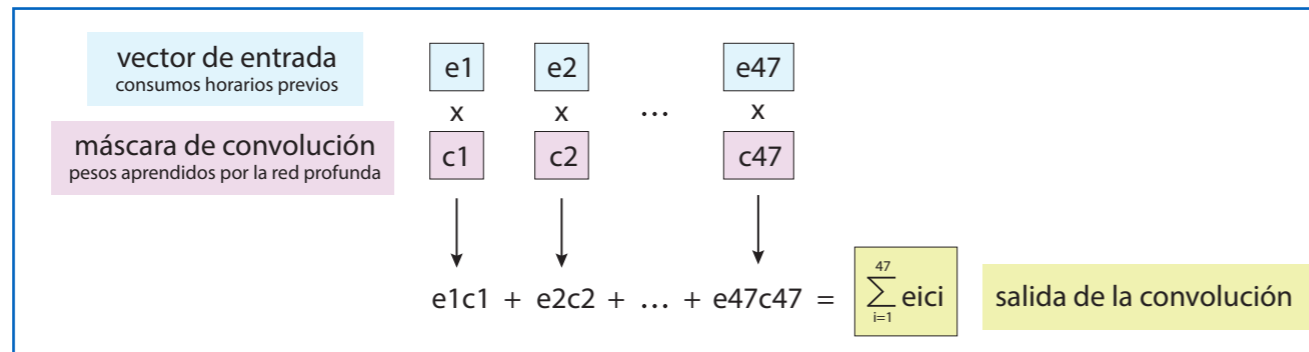


Fuente

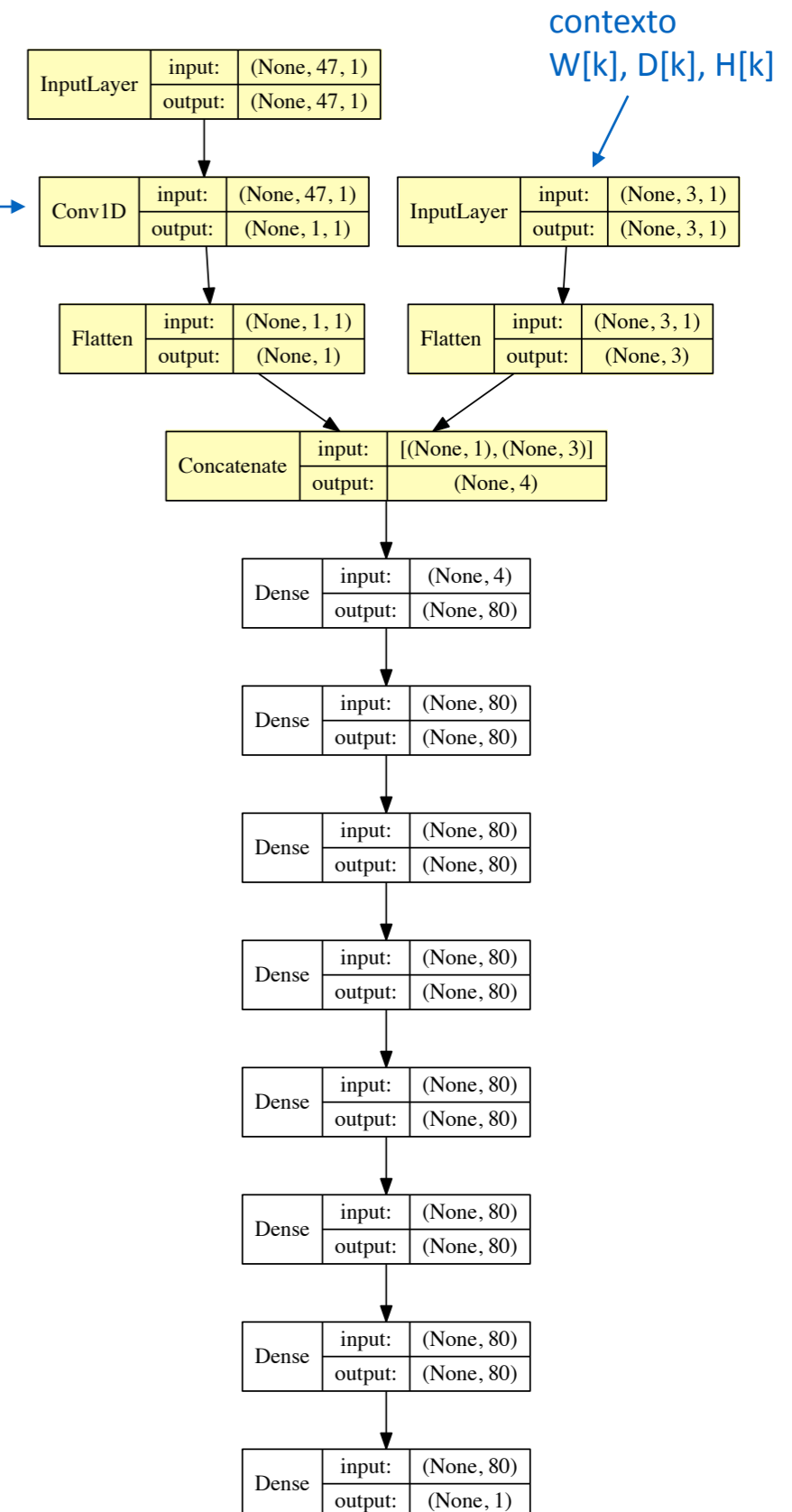
Ana González Muñiz, "Aplicaciones de técnicas de inteligencia artificial basadas en aprendizaje profundo (deep learning) al análisis y mejora de la eficiencia de procesos industriales". Trabajo Fin

de Máster Universidad de Oviedo. Director, Ignacio Díaz Blanco.
de Conocimiento en Procesos de Ingeniería

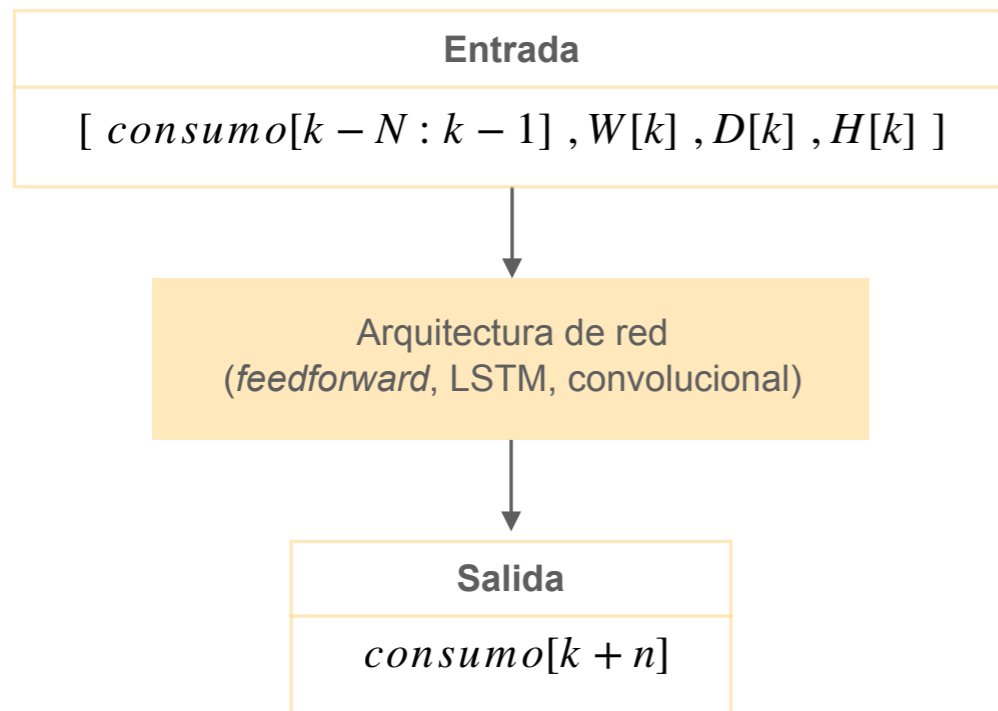
Convolución con ventana de 48h de consumos previos



Arquitectura convolucional



Planteamiento



Fuente

Ana González Muñiz, "Aplicaciones de técnicas de inteligencia artificial basadas en aprendizaje profundo (deep learning) al análisis y mejora de la eficiencia de procesos industriales". Trabajo Fin

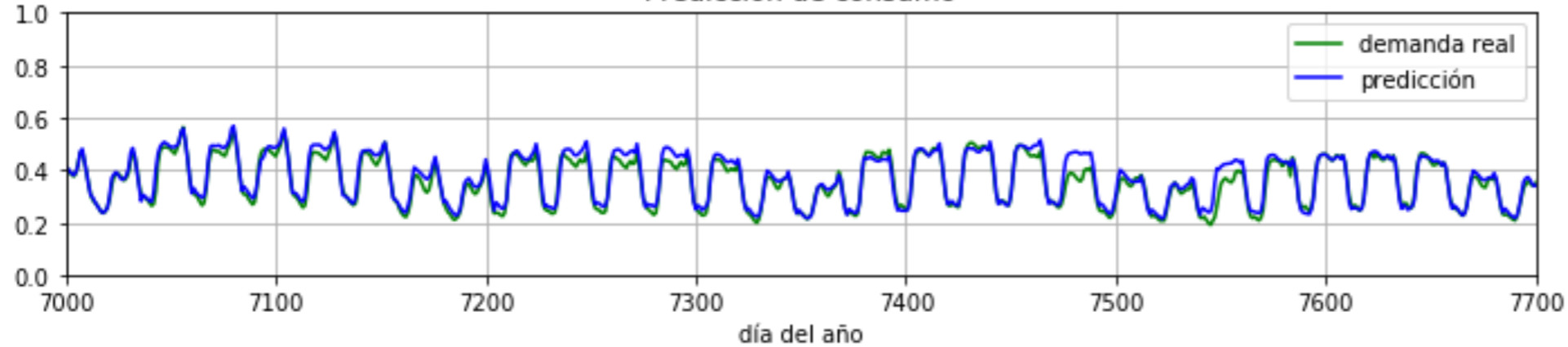
de Máster Universidad de Oviedo. Director, Ignacio Díaz Blanco. de Conocimiento en Procesos de Ingeniería

Aplicaciones de DL

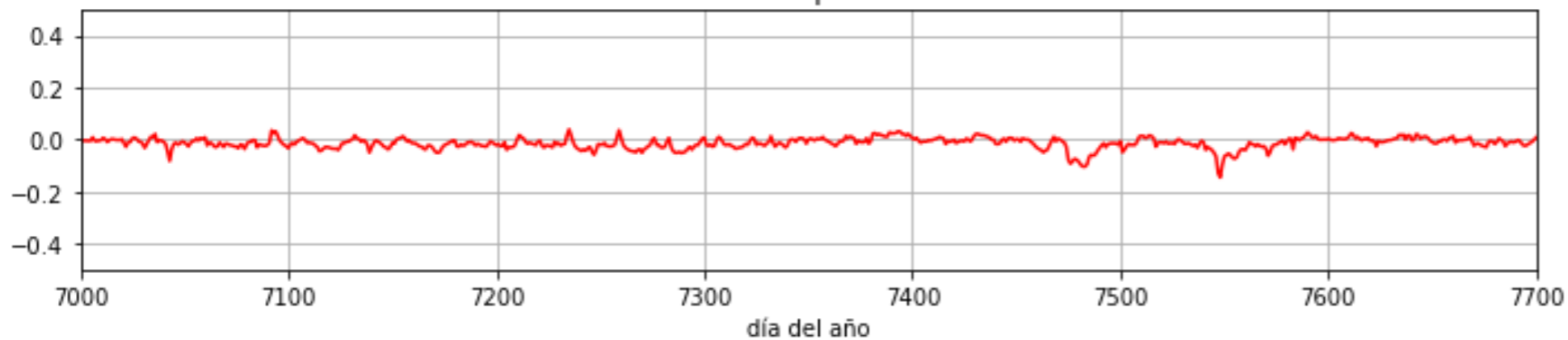
Predicción de demanda eléctrica

Resultados

Predicción de consumo



Error de la predicción



Comparación con otros métodos

	Error cuadrático medio	
	Entrenamiento	Test
Red <i>feedforward</i>	6.88×10^{-4}	9.04×10^{-4}
Red LSTM	5.55×10^{-4}	5.99×10^{-4}
Red convolucional	5.90×10^{-4}	6.00×10^{-4}
Modelo lineal (Ridge Regression)	29.83×10^{-4}	31.51×10^{-4}
SVR (Kernel rbf)	5.71×10^{-4}	7.58×10^{-4}

Fuente

Ana González Muñiz, "Aplicaciones de técnicas de inteligencia artificial basadas en aprendizaje profundo (deep learning) al análisis y mejora de la eficiencia de procesos industriales". Trabajo Fin de Máster, Universidad de Oviedo, Director, Ignacio Díaz Blanco.

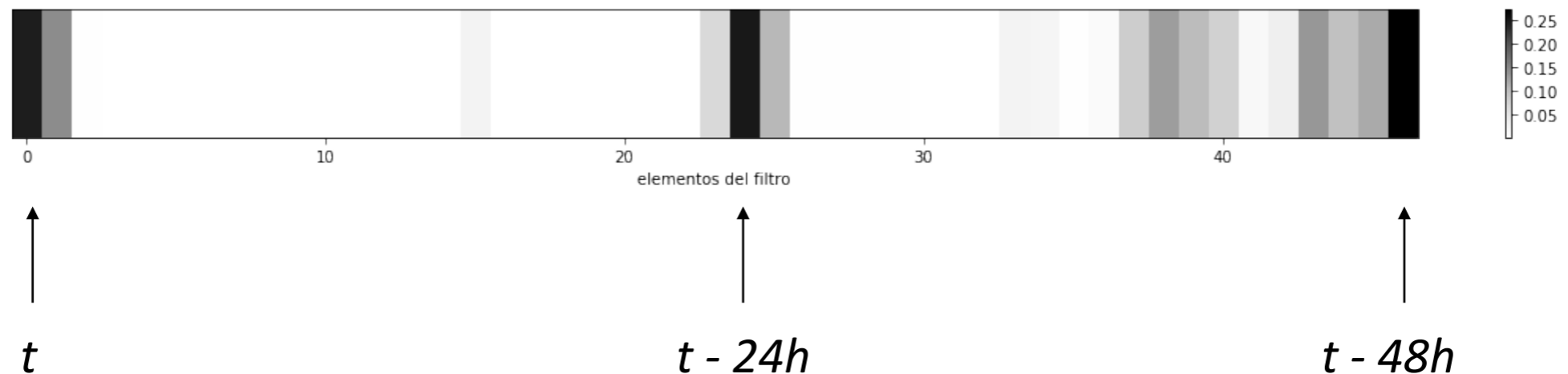
Aplicaciones de DL

Predicción de demanda eléctrica

Resultados

Aprendizaje de la representación

Pesos de las 48h anteriores
asignados por la capa convolucional



la red fue, además, capaz de determinar por sí misma
qué horas son relevantes en la predicción
del consumo 24h después

Fuente

Ana González Muñoz, "Aplicaciones de técnicas de inteligencia artificial basadas en aprendizaje profundo (deep learning) al análisis y mejora de la eficiencia de procesos industriales". Trabajo Fin

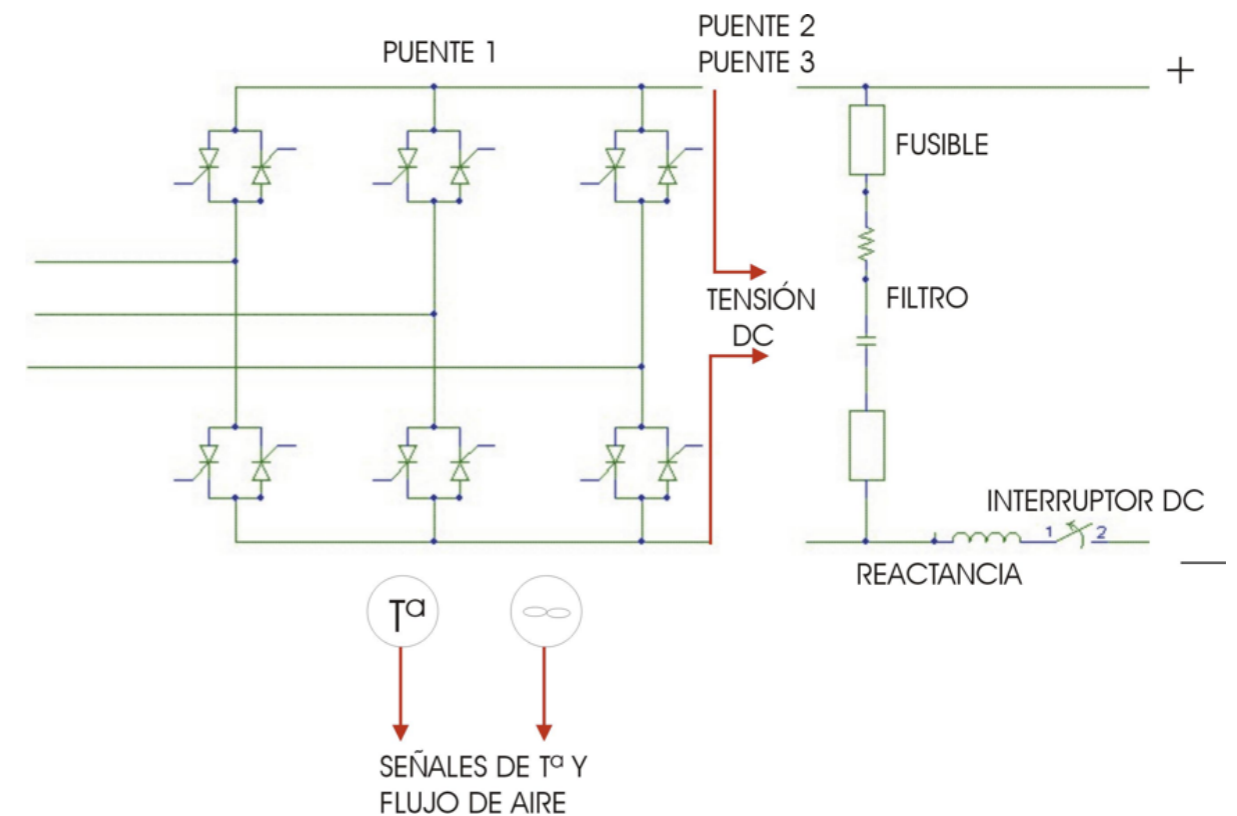
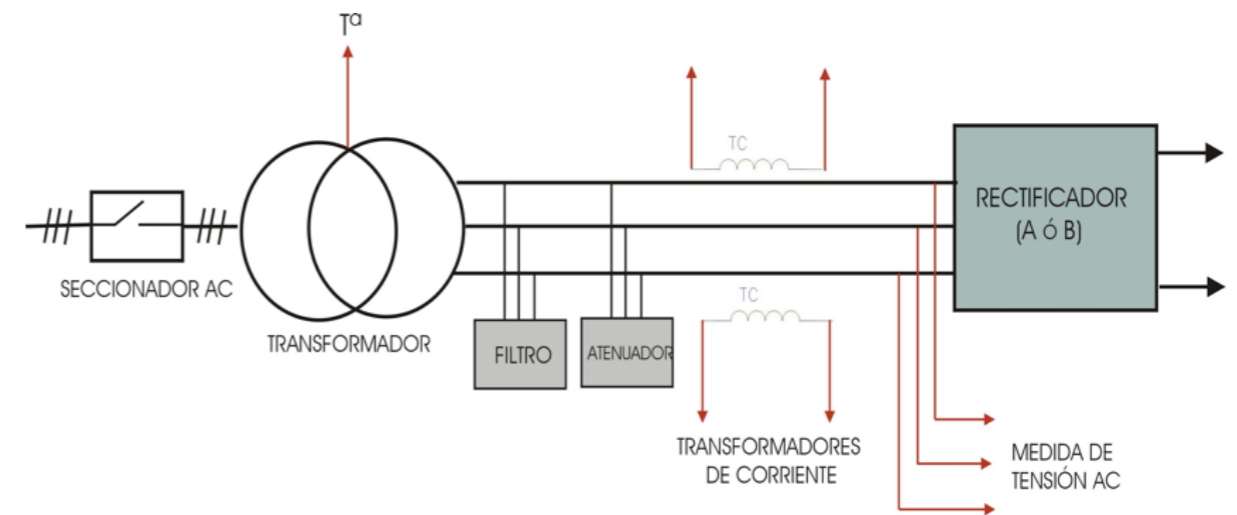
de Máster Universidad de Oviedo. Director, Ignacio Díaz Blanco.

de Conocimiento en Procesos de Ingeniería

Aplicaciones de DL

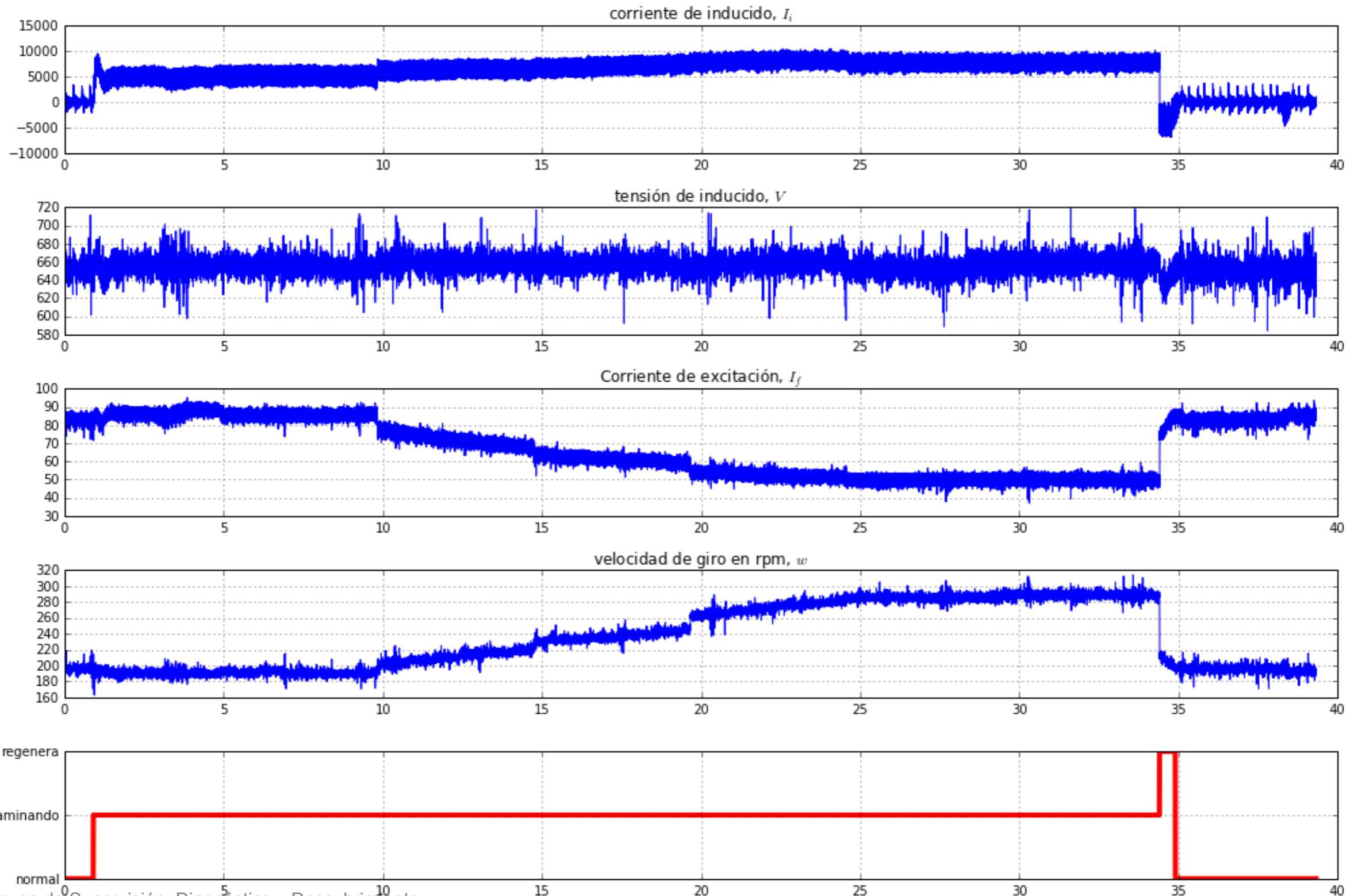
Análisis de corrientes en motor DC

Ficha Técnica	
Archivo	<ul style="list-style-type: none"> datosmotorlaminacion.zip
Descripción	Variables del motor antes, durante y después de una laminación en el TBC. El motor en cuestión ya no existe y ha sido reemplazado en una remodelación del TBC.
Características del motor	<ul style="list-style-type: none"> Motor CC de excitación independiente Potencia: 6000 kW Tensión de inducido nominal: 700 V Corriente de inducido nominal: 9000 A Corriente de campo nominal: 150 A Velocidad nominal: 180 rpm Velocidad máxima: 360 rpm Localización: Tren acabador en caliente Caja: F2



Aplicaciones de DL

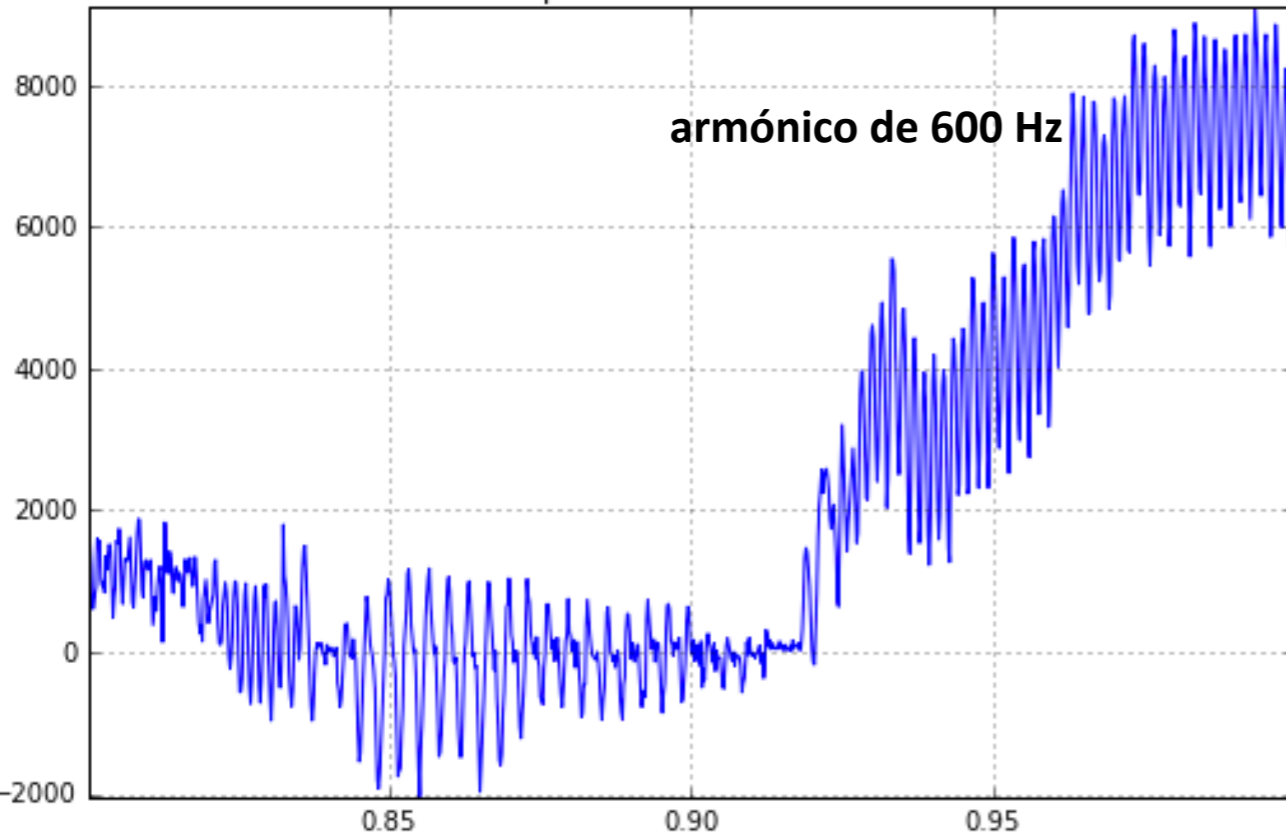
Análisis de corrientes en motor DC



Aplicaciones de DL

Análisis de corrientes en motor DC

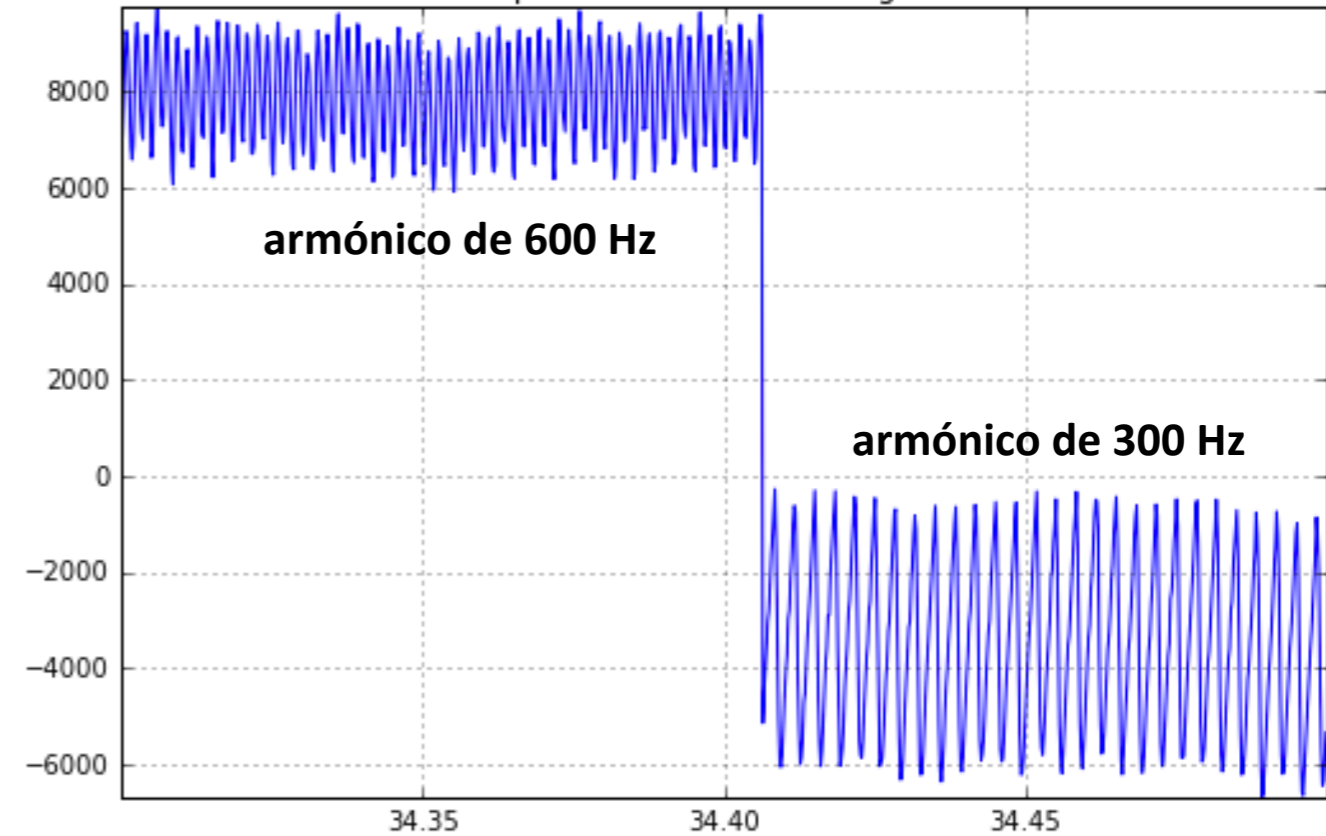
detalle (paso de vacío a laminando)



vacío

laminando

detalle (paso de laminando a regeneración)

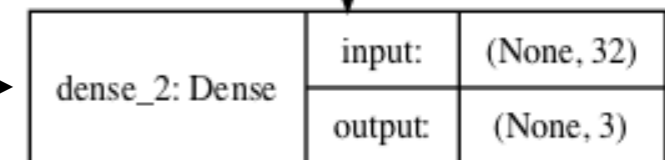
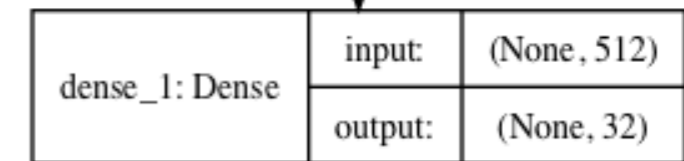
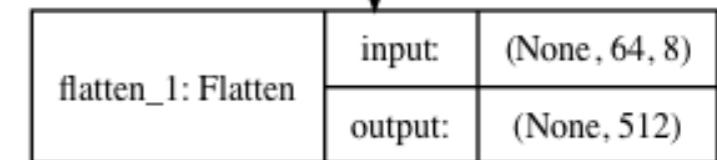
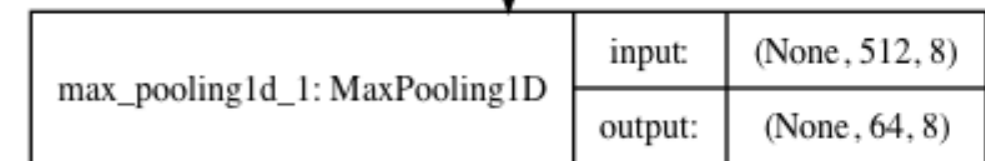
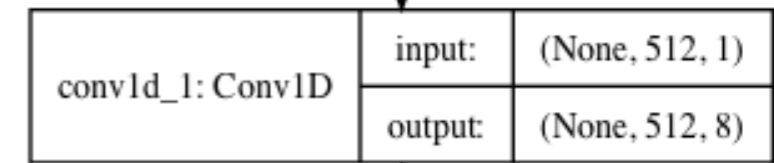
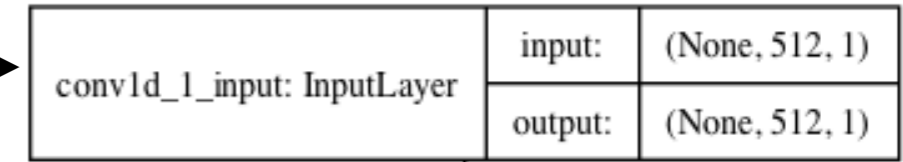
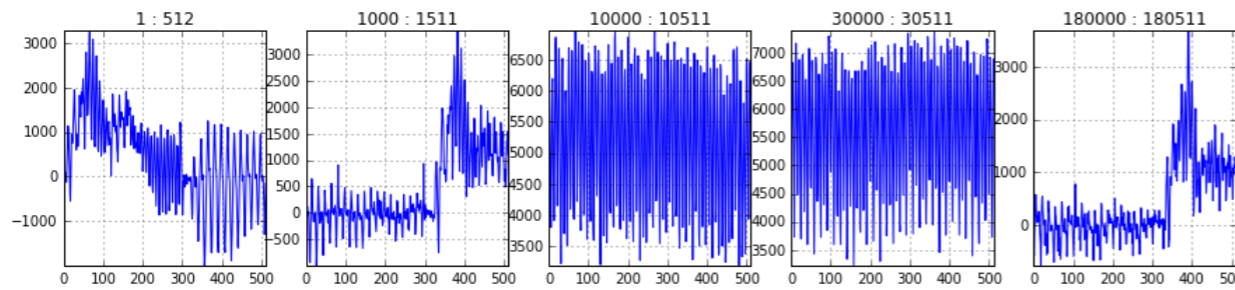


laminando

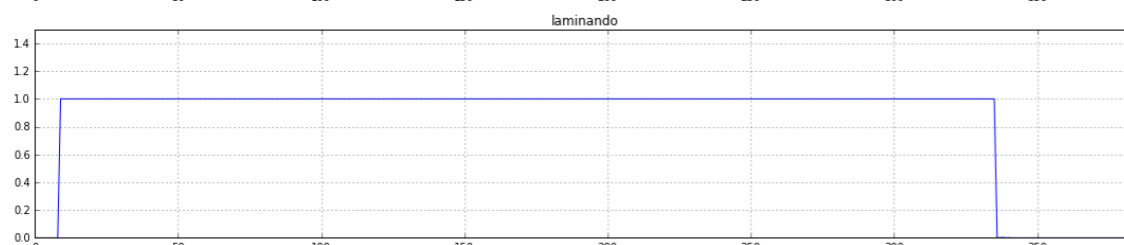
regeneración

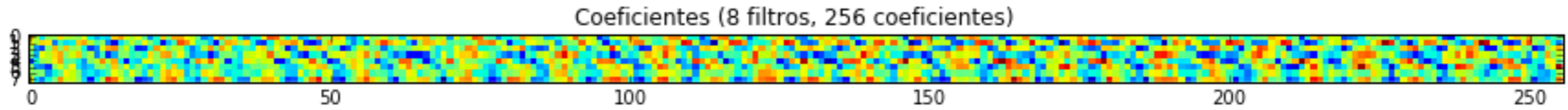
Análisis de corrientes en motor DC

Ventanas de datos de entrada

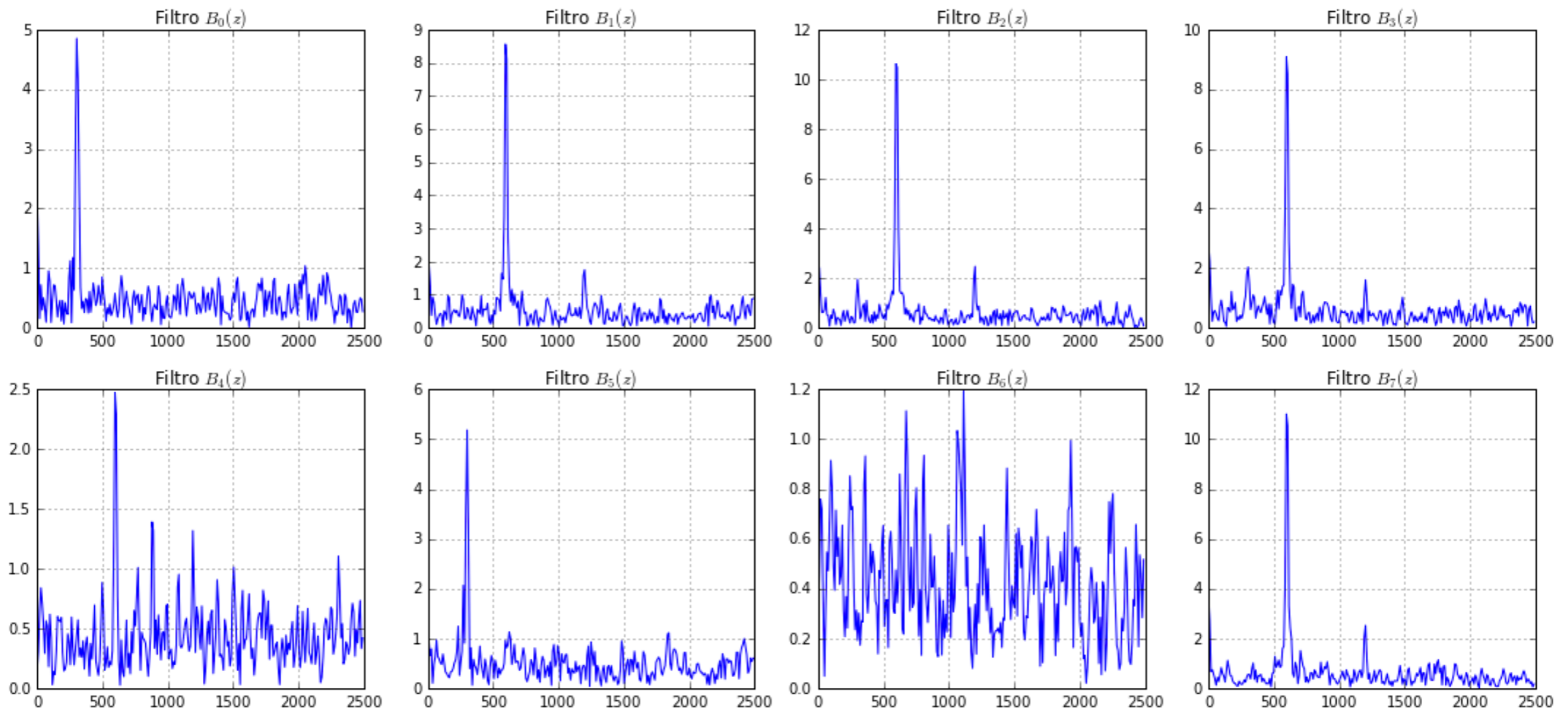


Clases ("one-hot" encoding)





Respuestas en frecuencia de los 8 filtros



Ejemplo “de juguete” en TF/Keras

clasificación 2D

```
import matplotlib

from pylab import *
from keras.models import Sequential
from keras.layers import Dense, Activation
from keras.optimizers import SGD
from keras.utils.np_utils import to_categorical
%matplotlib inline
```

Datos de entrenamiento

```
# creamos varias nubes de puntos 2D
X_train = vstack((randn(100,2)+array([-3,2]),
                  randn(100,2)-3,
                  randn(100,2)+array([0,4]),
                  randn(100,2)+array([-5,-6]),
                  randn(100,2)+array([0,-6])))

# las primeras son clase "1" y las restantes serán de clase "0"
Y_train_classes = kron([1,1,0,0,0],ones(100))

# generamos una codificación tipo "one-hot" (vectores de unos y ceros: un elemento para cada clase)
Y_train = to_categorical(Y_train_classes).squeeze()

Y_train.shape
```

In [8]:

```
(500, 2)
```

Out[8]:

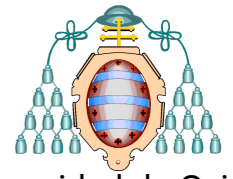
Datos de test

```
# creamos una retícula que recubra todo el espacio
# para comprobar los resultados de la clasificación en puntos intermedios
xi = linspace(-10,10,50)
xx,yy = meshgrid(xi,xi);
X_test = vstack((xx.flatten(),yy.flatten())).T
```

In [9]:

Ejemplo “de juguete” en TF/Keras

clasificación 2D



Creamos el modelo (varias capas)

In [10]:

```
# creamos el modelo: varias capas
model = Sequential()
model.add(Dense(units=64, input_dim=2))
model.add(Activation("relu"))
model.add(Dense(units=64, input_dim=2))
model.add(Activation("relu"))
model.add(Dense(units=2))
model.add(Activation("softmax"))

# usamos "categorical_crossentropy", aporta más sensibilidad en problemas de clasificación
model.compile(loss='categorical_crossentropy', optimizer=SGD(lr=0.01, momentum=0.9, nesterov=True))

# ajustamos el modelo empleando el optimizador y función de coste anteriores
model.fit(X_train, Y_train, nb_epoch=50, batch_size=32, verbose=1)
```

```
Epoch 1/50
500/500 [=====] - 0s - loss: 0.4670
Epoch 2/50
500/500 [=====] - 0s - loss: 0.2924
Epoch 3/50
500/500 [=====] - 0s - loss: 0.2487
Epoch 4/50
500/500 [=====] - 0s - loss: 0.2237
Epoch 5/50
500/500 [=====] - 0s - loss: 0.2048
Epoch 6/50
500/500 [=====] - 0s - loss: 0.1916
Epoch 7/50
500/500 [=====] - 0s - loss: 0.1790
Epoch 8/50
500/500 [=====] - 0s - loss: 0.1679
Epoch 9/50
500/500 [=====] - 0s - loss: 0.1577
Epoch 10/50
500/500 [=====] - 0s - loss: 0.1531
Epoch 11/50
500/500 [=====] - 0s - loss: 0.1436
Epoch 12/50
500/500 [=====] - 0s - loss: 0.1316
Epoch 13/50
500/500 [=====] - 0s - loss: 0.1305
Epoch 14/50
500/500 [=====] - 0s - loss: 0.1299
```

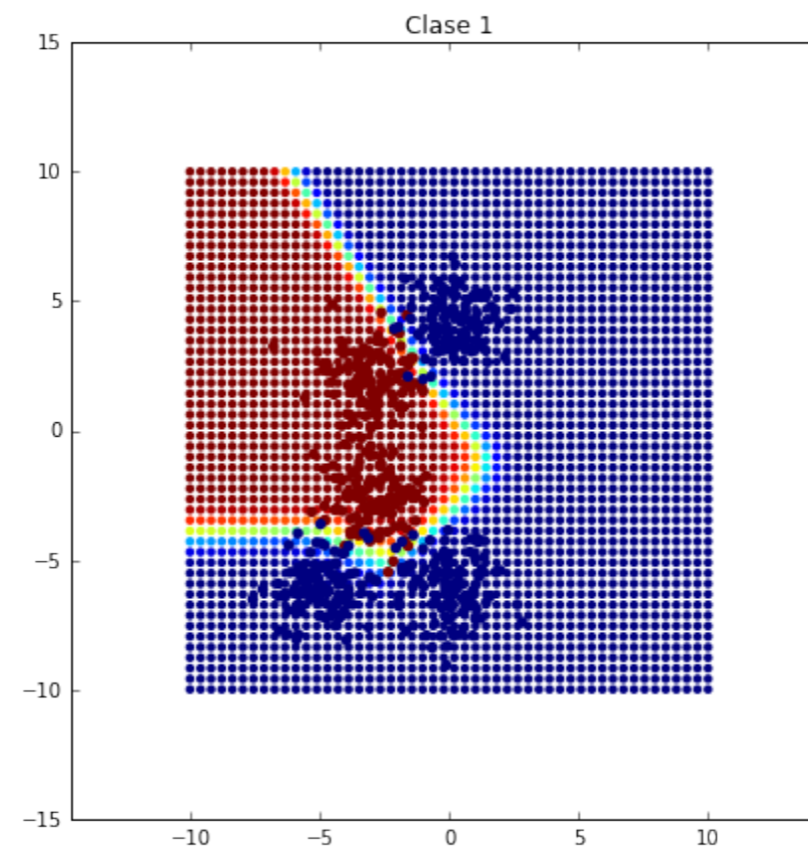
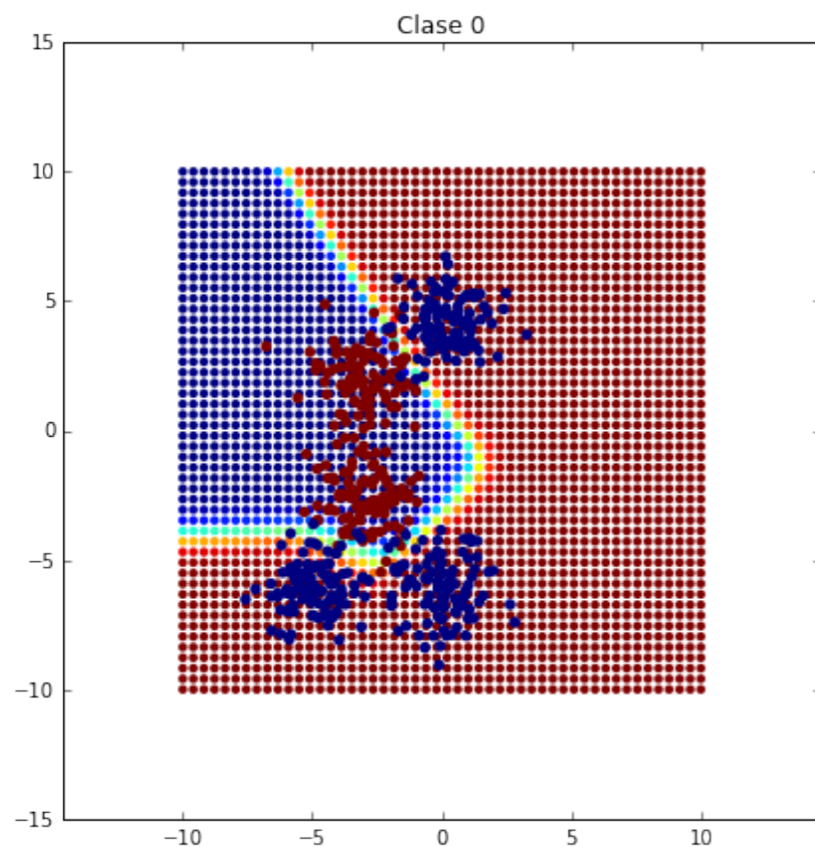
Predicción

In [11]:

```
# predicción de las clases
classes = model.predict_classes(X_test, batch_size=32)

# predicción de las probabilidades de cada clase
class_probabilities = model.predict_proba(X_test, batch_size=32, verbose=0)

# dibujamos los resultados
figure(1, figsize=(15, 7))
clf()
for i in range(2):
    subplot(1, 2, i+1)
    scatter(X_test[:, 0], X_test[:, 1], c=class_probabilities[:, i], linewidths=0)
    title('Clase %d'%i)
    axis('equal')
    scatter(X_train[:, 0], X_train[:, 1], s=30, c=Y_train_classes, linewidths=0)
```



In []:

<http://isa.uniovi.es/GSDPI>

Inspección superficial

- Weimer D, Scholz-Reiter B, Shpitalni M. Design of deep convolution neural network architectures for automated feature extraction in industrial inspection. CIRP Ann Manuf Technol 2016;65(1):417–20.
- Ren R, Hung T, Tan KC. A generic deep-learning-based approach for automated surface inspection. IEEE Trans Cybern 2017;99:1–12.
- Masci J, Meier U, Ciresan D, Schmidhuber J, Fricout G, Mittal A. Steel defect classification with max-pooling convolution neural networks. IEEE international joint conference on neural networks (IJCNN) 2012;20:1–6.

Detección e identificación de fallos

Cajas de cambios

- Chen ZQ, Li C, Sanchez RV. Gearbox fault identification and classification with convolution neural networks. Shock Vib 2015;2:1–10.

Cojinetes

- Shao H, Jiang H, Zhang X, Niu M. Rolling bearing fault diagnosis using an optimization deep belief network. Meas Sci Technol 2015;26(11):1–17.
- Gan M, Wang C, Zhu C. Construction of hierarchical diagnosis network based on deep learning and its application in the fault pattern recognition of rolling element bearings. Mech Syst Signal Process 2016;72–73(2):92–104.

Trenes

- Yin J, Zhao W. Fault diagnosis network design for vehicle on-board equipments of high speed railway: a deep learning approach. Eng Appl Artif Intell 2016;56:250–9.

Maquinaria (predicción de desgaste, caja de cambios, cojinetes)

- Zhao R, Wang D, Yan R, Mao K, Shen F, Wang J. Machine health monitoring using local feature-based gated recurrent unit networks. IEEE Trans Ind Electron 2018;65(2):1539–48.

Predicción de demanda eléctrica

- Qiu X, Zhang L, Ren Y, Suganthan PN, Amaratunga G. Ensemble deep learning for regression and time series forecasting. IEEE symposium series on computational intelligence 2014:1–6.

Fin de la presentación